Microprocessor-based digital flight control system design for an R.P.V.

Gittens, Simon Nevis

Award date:
1985

Awarding institution:
University of Bath

Microprocessor-based digital flight control system design for an R.P.V.

Submitted by:

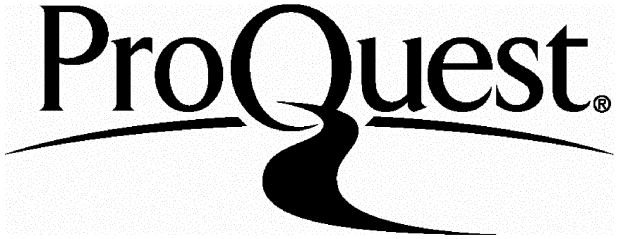
Simon Nevis Gittens B.Sc. (Hons.)

for the degree of Ph.D.

of the University of Bath.

1985

Bath, May 1985



Summary.

The development of a microprocessor based digital flight control system for a particular R.P.V. is described. The tasks required of this system are defined, and thereafter, the hardware circuits and the software structure necessary to implement a prototype are presented. The autopilot control laws are inferred from z-plane root loci, and then confirmed using digital simulations of the de-coupled roll and pitch attitude loops. The problems of the finite wordlength implementation of the control laws are discussed, and then both hybrid simulation and actual flight results are used to prove the performance of the prototype.

To exploit the adaptive capabilities of a software based system, a sliding mode variable structure control law is developed for the roll attitude loop. Digital simulations are used to show that significant improvements in sensitivity reduction can be achieved under some conditions. These improvements are lost if a realistic servo-actuator model is employed. Another objective, namely the reduction of the disturbance error induced by trim imbalance, is maintained provided a reduced order switching function is used.

Acknowledgements.

The work presented in this thesis was carried out under the supervision of Mr. A.R. Daniels, Senior lecturer in the School of Electrical Engineering, University of Bath, England. The author wishes to express his gratitude to Mr. Daniels for his interest, understanding and advice in many matters.

The author is also grateful to Dr. B.A. White, lecturer, for his frequent guidance on Variable Structure System theory, and the entire of the R.P.V. Research department at British Aerospace for the practical assistance they have given. Financial support from the Science and Engineering Research Council, and the Training department, British Aerospace, Filton, is gratefully acknowledged.

The author is indebted to Professor J.F. Eastham for the facilities provided by the University.

CONTENTS.

SYMBOLS AND ABBREVIATIONS.

Notation used in the text:

SENS, TIBUG	Refers to a software label, which might be a single word, a routine or a complete programme.
"BATOK", "RESET"	Refers to a named signal in the hardware system.
(nn)	The number applies to the list of references.
°NNNN°	The expression between the delimiters is a binary number, written in hexadecimal notation.
g(t)	Analogue, continuous function of time.
g*(t)	Analogue, impulse sampled function of time.
g(kT)	Digital, sampled data sequence. (Where T is the sampling period.)
$\mathcal{Z}\left[\quad\right]$	Z transform of the function inside the parentheses.
C(s)	Function of s.
$C_R(s)$	T.F. between the aileron deflection and the roll rate.
$C_P(s)$	T.F. between the elevator deflection and the pitch rate.
$C_{SV}(s)$	T.F. of linearised, first order model of the servo-actuator.
$C_C(s)$	T.F. of the pitch compensator.
G(z)	Function of z.
$G_R(z)$	T.F. between the aileron deflection and the roll attitude, including zero order hold.
$G_P(z)$	T.F. between the elevator deflection and the pitch attitude, including zero order hold.

$G_{SV}(z)$	T.F. of linearised, first order model of servo-actuator, including zero order hold.
$G_{RSV}(z)$	T.F. between the aileron demand and the roll attitude, including zero order hold.
$G_{PSV}(z)$	T.F. between the elevator demand and the pitch attitude, including zero order hold.
$G_C(z)$	T.F. of pitch compensator, with equivalent root positions to C (s).

Abbreviations:

A/D	Analogue to digital, (conversion).
A.T.F.	Aircraft transfer function.
CMOS	Complementary-metal oxide-on-silicon (logic).
c.p.u.	Central processing unit.
CRU	Communications register unit, (processor I/O).
D/A	Digital to analogue, (conversion).
DFCS	Digital flight control system.
EPROM	Eraseable, programmable read only memory.
HI	Logic high, or "1".
I/O	Input/output, (w.r.t. processor).
LO	Logic low, or "0".
Pn	I/O port n.
RAM	Random access memory.
Rn	Register n , or Resistor n. Refer to context.
R.P.V.	Remotely piloted vehicle.
s	Complex variable, (Laplace analysis).
TTL	Transistor-transistor logic.

Abbreviations, (cont.):

V.S.S.	Variable structure system.
z	Complex variable (sampled data analysis).

Symbols:

Symbols in general use (m, Hz, A, V, rad, etc.) will not be defined. Lower case, bold type face letters denote a vector (e.g. $\mathbf{a}$), upper case, bold type face letters denote a matrix*(e.g. $\mathbf{A}$). The suffix $\mathbf{a}^t$ denotes the transpose of the vector or matrix. "Dot" notation is used to denote derivatives of a function (e.g. $\dot{y}$, $\ddot{y}$). "Sgn(fn)" denotes the sign of the function fn.

f	Disturbance input.
s	Complex variable, (Laplace analysis).
T	Sampling period.
u	control signal.
x	System state, differential equation.
X	System state, difference equation.
X (ss)	Steady state value of system state.
z	Complex variable, (sampled data analysis).
m	Linear operator in n-th order vector space specifying switching function.
	Fixed part of gain vector.
$\Delta \mathbf{K}$	Switched part of gain vector.
ϕ	Roll attitude.
θ	Pitch attitude.
ψ	Yaw attitude. (Part II)

* the execption to this is $\mathbf{K}$ which is used to denote a gain vector.

Symbols, (cont.):

ψ	(Part III) COefficient capable of switching from one value to another, (also known as "key" function).
σ	Switching function.
λ	Eigenvalue
ζ	Aileron deflection.

Chapter 1 Introduction.

1.1 Introduction to Remotely Piloted Vehicles.

Within eleven years of the Wright Brothers' first flight in a powered aircraft in 1903, work began to reverse the process and develop functional un-manned aircraft. The advent of air warfare compounded the risks in what was already a hostile environment, so that in 1914 the War Office asked Professor A.M.Low to design a remotely controlled pilot-less aircraft (1). This was originally intended as a flying bomb for use against Zeppelin airships and as such was more of a missile than that which today is called an R.P.V.(Remotely Piloted Vehicle). This particular aircraft was never completed, but later projects by Low et al led to the first successful flight of an R.P.V. in 1924, and the first mass produced R.P.V. This was known as the Queen Bee, of which 420 were built between 1934 and 1943.

Though no rigid nomenclature exists, the term R.P.V. has now come to mean a specific type of pilot-less vehicle. In attempting to define a modern R.P.V. consideration must be given to the manner in which it is controlled, and the purpose for which it is intended. A suitable definition might be; that an R.P.V. is an aircraft that is controlled directly, but remotely by a human pilot, combining commands received at some time during the flight with on-board sensor information to achieve varying degrees of automatic control. It is not usually considered to be expendable, and so some means of recovery will be provided.

The enthusiast's model aircraft is excluded because it carries no

sensors, as are missiles which clearly are not recoverable. Furthermore, if the aircraft is totally autonomous, performing a fixed flight routine, it cannot be considered as a true R.P.V. as it does not receive commands while in flight. Such a vehicle is called a Drone (2).

Most aerial targets, flying bombs and decoys are operated in drone form, whereas true R.P.V.s are more commonly used in situations where interaction with the operator is required. This usually involves some form of reconnaissance. To date, sadly, all R.P.V. development has been for military purposes. They are, however suitable for certain civil roles being particularly effective for low speed, low altitude work. Traffic monitoring, transmission line checking, search operations and even crop spraying might be achieved with cost savings over a manned light plane.

The aircraft for which this work was carried out is the Stabileye Mk 3, a mini R.P.V. designed and produced by the R.P.V. Technology Group, British Aerospace PLC. This is a general purpose payload carrying vehicle as shown in Fig. 1. Stabileye is used to provide an airborne platform for customer's electronics in situations where conventional aircraft would be unsuitable. It can be operated in a variety of modes, ranging from direct pilot command of the control surfaces to a completely automatic sequence of manoeuvres. The aircraft will be described in more detail in Chapter 2.

1.2 The application of microprocessors to R.P.V. control.

The purpose of this work was to develop a microprocessor based flight

control system for the R.P.V. All previous Stabileye autopilots have employed analogue control loops using a combination of operational amplifiers, analogue switches and discrete logic. However the transmission systems to and from the ground station, referred to as telecommand and telemetry, and the servomotor driving functions, are all of a digital nature. Therefore, much of the circuitry was required to perform analogue-to-digital and digital-to-analogue conversions. While these methods are adequate, they become very complex as the number of operational modes increases. Furthermore, many of the anticipated developments in R.P.V. technology require data storage of some form, such as pre-programmed flight plans, for which the conventional system would require considerable modification.

The advent of low cost microprocessor technology has provided the means to produce a digital computer autopilot system, smaller than its analogue counterpart, for comparable cost. Such a system would require less signal conversion hardware and would have its own data storage medium. Of more importance, however, are the advantages to be had by defining the control law in software. The system becomes far more flexible, (attractive for a low volume research application), less susceptible to variation through aging and temperature effects, and more easily repeated from one unit to another. Also, adaptive control strategies are much easier to implement.

A microprocessor based system design is dependent on the trade-off of hardware circuitry and software programming such that, as a general rule ,the more complex one is, the simpler the other becomes. Clearly to obtain maximum benefit, the hardware should be as simple as

possible, while maintaining an acceptable margin of safety in the percentage usage of c.p.u. time.

In this application, the optimum solution would be one that treated the control, telecommand, and telemetry systems as a whole, using software methods for all encoding and decoding operations. However, the need to maintain low overall development costs required that the telecommand, telemetry and sensors should remain unaltered from the existing analogue scheme. Hence the equipment described here is intended as a first prototype only, to prove the viability of a microprocessor based flight controller.

1.3 Development of the control laws.

The development of the control loops first required that the dynamics of the aircraft with respect to the control surface deflections be defined. The equations of motion are derived from wind tunnel measurements, which are simplified using certain assumptions to give the decoupled, fixed velocity, short period transfer functions that are used in the control analysis.

The digital autopilot is necessarily a sampled data design problem which involves, either, digitisation of an existing analogue control law or redesign in the discrete domain. The latter method was chosen, so the aircraft transfer functions had to be converted to a suitable form to allow the gains and compensators to be defined directly. This transformation process was carried out, in part, using matrix and polynomial solution packages on a mainframe computer system, to give the z transforms of the transfer functions and the z plane root loci.

These were used to suggest the type of control, and then the gains were confirmed by checking the step responses at various airspeeds in digital simulations of the loops. Constraints in the position and rate of the controlling member were used at this stage.
The next process was tb prove the controller as a complete unit in the presence of system non-linearities and noise sources. This was done using a hybrid simulation, whereby the flight control electronics were connected to analogue computer models of the aircraft transfer functions. In this way, the effect of coefficient quantisation and other implementation errors could be examined. The final stage in the development of the fixed gain control laws was to prove the system in a flight trial.

Parts 1 and 1l of this thesis deal with the development of the digital flight control system up to the point where it is flight tested with a fixed structure control law. As an extension of this, part 111 investigates the possibility of adaptive control. This is quite separate from the material in the first two parts, and will not be described here. An introduction to the particular form of adaptive control used, namely sliding-mode control, is given in Chapter 8.0.

Part I DESCRIPTION OF THE SYSTEM.

Chapter 2 Introduction to the Flight Control System.

2.1 The aircraft.

A general view of the aircraft is given in Fig. 1. The flying surfaces are constructed from polystyrene foam with wood veneer skin while the fuselage, booms and all necessary strong points are made of glass reinforced plastic in various forms. The take-off weight is 64kg, of which 25kg are attributed to payload or ballast. Thrust is provided by a 25 h.p., twin cylinder two-stroke engine driving a pusher propellor, and electrical power is provided by batteries. This configuration is chosen to allow unobstructed forward view, desirable for its primary function as a reconnaissance platform. Typically the aircraft will operate with airspeeds between 22 and 50m/s. The stall speed is approximately 17m/s with a nominal cruise speed of 40m/s. The payload is located in the forward compartment, while the instrumentation pack and fuel tank, which should be at the approximate centre of gravity of the aircraft, are placed in the mid and rear compartments respectively. The airframe was designed to have good natural stability in all planes for satisfactory, unassisted pilot control. Elevator and rudder deflections are linear but the ailerons have differential action provided mechanically by an offset crank.

The aircraft is launched using a pneumatic ram and recovered by deploying a parachute. The risk of damage on landing is reduced by the use of a cushioning airbag.

2.2 The ground station.

The complete R.P.V. system consists of the ground equipment and one or more aircraft. In its simplest form, the ground station is composed of a telecommand transmitter, with some form of pilot control unit, and a telemetry receiver with a means of displaying important information. If the aircraft are operated beyond line of sight then a method of pinpointing their position is required. Usually, radar tracking is used, but real time video images might be adequate for this purpose, if the operator is familiar with the area. The exact configuration of the ground station depends upon the payload and the purpose of the mission.

Full details of the telecommand (3) and telemetry (4) systems are beyond the scope of this report. However a brief description of their data structures will be given as these are directly related to the functions of the flight control system.

The telecommand transmission passes information to the aircraft. This will include demands for the control loops, payload manipulation and flight mode selection data. Up to 16, 8 bit channels are available with the following definitions:

Channel 0 - null, no new data

Channel 1 - roll data

Channel 2 - pitch or height data

Channel 3 - yaw or heading data

Channel 4 - throttle or airspeed data

Channel 9 - payload command word

Channel 10 - command word

The remaining channels are unspecified and can be configured for payload use.

The transmission is continuous with pitch and roll information being broadcast alternately, whereas other functions are only broadcast when they change in value. As the order of transmission is random, an 8 bit address word is required to define the channel number and the aircraft for which the information is intended. Each command frame of 48 bits is composed of a synchronisation word of 16 bits, an 8 bit combined vehicle and channel address with its shuffled complement, and an 8 bit data word with its shuffled complement. The bit rate is 2048 baud so the aircraft receives a new command frame approximately 43 times a second. When no new information is available or when the pilot control unit has been de-activated, the null channel can be broadcast to maintain the link and prevent the flight control system initiating a failsafe routine.

Telemetry is the process of returning information to the ground station to provide a real time record. This data will include sensor measurements, actuator demands and system status. Up to 24 analogue and 6 digital channels are catered for, compiled into 32, 8 bit words as shown in Fig. 2, (channels 1 and 2 are used as a synchronisation word). These are broadcast in numerical order, the data rate being 4096 baud, so that the sampling period of any particular channel is 0.0625s. Full details of the assignment of the telemetry channels for this system are given in Appendix A.

2.3 Modes of operation.

There are four modes of flight, each representing a different degree

of R.P.V. autonomy. These are, direct pilot control, autopilot, avionics and failsafe, the latter being a fixed sequence of manoeuvres performed in the event of loss of the telecommand link. The flight mode is selected by bits in channel 10, the command word, which dictates the interpretation of the information in the pitch, roll and yaw channels. The meaning of each bit in the command is given in Fig. 3.

In the direct pilot mode, the pitch, roll and yaw information commands the position of the control surfaces in the same way as a model aircraft. To be of practical use, the R.P.V. must be within line of sight. This mode requires a greater level of skill from the operator and would only be used if there was reason to doubt the performance of the autopilots, such as during preliminary flight trials or in the event of instrument failure. Direct pilot command of the throttle, parachute deployment, airbag deployment and ignition cut facilities are also provided.

Selecting the autopilot mode closes the pitch and roll control loops, so that the values in channels 1 and 2 are now interpreted as desired attitudes. The elevator and aileron deflections are calculated in control laws which use measurements from a dual axis attitude gyro to provide feedback information. A choice of algorithms is available, fixed gain replicas of the existing analogue control laws, and secondly, variable structure routines. The sign conventions used are given in Chapter 5.

The third flight mode is the avionics mode which incorporates the pitch autopilot described above in a height-lock controller. In the height-lock mode, information derived from a barometric sensor is used to attempt to maintain the aircraft at a desired height above a

previously defined zero level. The height demand is specified in the pitch channel. The throttle is under direct pilot control and must first be adjusted to provide adequate thrust to allow the aircraft to climb if necessary. The pitch autopilot limits the attitude, so that, for a large height error and a fixed throttle setting, a reasonably constant rate of climb will be achieved.

The final flight mode is the fully autonomous failsafe mode which takes command of the aircraft in the event that the telecommand link is lost. An interruption in the link could occur because the aircraft has gone out of range, or due to a failure at the transmitter or receiver. There may also be interference from external r.f. emitters. The receiving unit uses synchronisation bytes and shuffled complement bytes to detect errors, but no attempt to correct them is made and erroneous data is simply rejected. Under these conditions, the aircraft will continue to perform the last command it received which means it is effectively out of control. Clearly this is undesirable so some automatic recovery scheme is required.

The action of the failsafe procedure is complicated by the presence of an analogue back-up flight control system, included until the digital system is proven in flight. This back-up has its own rudimentary failsafe procedure in addition to that provided by the software. The choice of analogue or digital control systems can be made via the command word, and, in the event of loss of telecommand, each system will provide its own failsafe.

If telecommand loss occurs during operation under analogue control, the analogue failsafe will be used. This provides the bare minimum of action necessary to bring the aircraft to the ground. When the telecommand fails, a delay of 2.5s will occur before the

parachute and airbag are deployed and the throttle is closed, thus shutting down the engine. The telecommand can resume control at any time during the routine. An oscillating bit in the telemetry indicates that the analogue failsafe has taken place. This signal, which has a period of about 5.0s, is also that used to operate the parachute and airbag. The routine will take place at any height and so the parachute may not necessarily have time to deploy properly when flying at low altitudes.

Alternatively, if the receiver detects loss of r.f. while under digital control, a flag is set in the telemetry to inform the ground station of the condition, and a software failsafe may be initiated. This is more flexible than a hardware based sequence, the only limiting factors, being the sensor information available, and the amount of programme storage given over to a failsafe routine. In the extreme, with navigational data available, the R.P.V. can continue with a pre-programmed mission, or return to the ground station before landing, thus reverting to the status of a fully autonomous drone. The failsafe software in this instance duplicates the procedure provided by the existing flight control system.

A timing diagram of the software failsafe is given in Fig. 4 and a description follows. If a delay of 0.5s has occurred since the last command, the autopilot and height lock are selected with a demand of 1000m, and full throttle is requested. This gives full pitch-up while roll and rudder demands are requested to make the aircraft climb in a spiral to 100m. This is to maintain the aircraft in approximately the position where contact was lost, while achieving a height which ensures safe deployment of the parachute. If the aircraft is above 100m, this phase is omitted. At this height, the roll attitude is set

to zero and a count of 5.0s is commenced. At any time up to the end of this period, the procedure can be halted, thereafter the throttle is closed, a further delay of 1.0s is observed, then the parachute and airbag are deployed. Any resumption of telecommand will reset the failsafe , so, if the link becomes intermittent and a failsafe is deemed necessary, the transmitter should be turned off by the operator.

While many different modes of control could be used (throttle to control height, elevator to control airspeed for example), the above were chosen as most of them duplicate facilities already proven with Stabileye. It is anticipated that a fully developed system would have a wide selection of flight modes from which those most suited to the payload could be taken for each individual mission.

2.4 Functions of the Digital Flight Control System, (DFCS).

The airborne electronics, which are based around a microprocessor system, are required to receive, interpret and perform commands, whether derived from the operator or the payload. This will involve gathering information, making decisions and then providing position information for the control surfaces. At all times, the security of the system must be maintained so a malfunction must be detected and acted upon. The functions of the electronics are thence as follows:

Receive and decode telecommand data

Gather information from sensors

Receive payload commands

Perform control laws

Drive control surfaces

Operate parachute etc.

Monitor system status for telemetry

Detect errors and take corrective action

Furthermore, as some of the tasks take place in dedicated hardware, and some in software, the correct synchronisation of the two is essential. A functional block diagram of the system is given in Fig. 5 showing which activities are performed by the processor. Each function will now be described more fully with details of hardware and software design given in Chapters 3 and 4.

2.4.1 System synchronisation and servomotor driving.

A sequential system, such as any single microprocessor based scheme, implies that activities be carried out one at a time, usually within a fixed cycle. It is necessary that the timing of the separate operations be arranged so as not to interfere with one another, and in such an order that all the information for an operation is available when required. Some activities must be carried out at a constant, predetermined rate, such as sensor measurements in a sampled data scheme, while others remain flexible. The timing of this system was dictated by the nature of the servomotors. These operate on pulse width modulated data provided at a fixed update rate. The pulse repitition period is 25ms so this was also chosen as the sampled data period. All system timing considerations such as delays, lapsed time checks etc. are achieved in software using multiples of this period, referred to as the System cycle.

The full range of a servomotor's movement is achieved by

supplying it with pulse widths varying from 0.9 to 1.9ms. By interleaving these pulses into the System cycle as shown in Fig. 6, eight servomotors can be driven by one driver circuit, thereby reducing the complexity of the hardware and spreading the current load. This leads to a sub-division of the System cycle referred to as the Servo period (=3.125ms). It is this period that is actually generated by the hardware, using a programmable count-down timer on the c.p.u. card, which produces synchronising interrupts, (level 3). The timing for the System cycle is determined by counting eight Servo periods. Hardware and software details for the servomotor driver are given in sections 3.2.3.3 and 4.4.4.1.

The telecommand and telemetry activities are provided by separate subsystems, each having their own timing, independent of the processor. Telecommand information is received approximately once per System cycle, the data being latched until the processor is ready to deal with it. Similarly, the telemetry is transmitted at somewhat less than once per cycle, so the processor stores the data in permanently enabled latches until such time as the telemetry encoder requires it.

2.4.2 Receiving and decoding telecommand information.

The receiver carries out serial-to-parallel conversion of the telecommand signal, extracts and validates the channel address and data, and then presents it at its output for a period of 7us. This information is stored until the processor is ready to deal with it. "New data ready" is signalled to the system using a low priority interrupt which is serviced once per System cycle. The act of reading

the telecommand port clears the interrupt flag, making it ready for the next command frame. All other decoding and data-directing is done by software.

The interpretation of this data depends on the state of certain bits in the command word which are monitored individually during the control routines. Certain functions such as parachute and airbag deploy, and ignition cut are confirmed by waiting for a given number of active commands before proceeding. Full descriptions of the hardware circuitry and software routines are given in sections 3.2.3.6 and 4.4.3.7.

2.4.3 The automatic control loops.

These all require an input demand and a feedback signal in order to produce some form of error value to drive the servo-actuators. All of the summing junctions, gains, compensators etc. are performed numerically in software, so that the only dedicated hardware components are the sensors and the means of reading them. The basic complement of instruments is as follows:

A dual axis attitude gyro (pitch and roll)

A rate gyro in the yaw plane (telemetry only)

A barometric altitude sensor

A barometric airspeed sensor (telemetry only)

Further details of these are given in Appendix B.

As most of the sensors produce analogue output signals, they have to undergo analogue-to-digital (A/D) conversion before being used in the control loops. This is achieved using an eight channel, multiplexed input A/D converter under the control of the software.

The A/D converter itself has a conversion time of 25us, which is fast enough to eliminate the need for a sample and hold circuit, and can be operated by simply delaying the processor during conversion without incurring a large time penalty.

The software routines all use fixed point, two's complement arithmetic with rounding and overflow correction where necessary. If compensators are required to achieve the desired specifications, these are accomplished using software implementations of digital filters. A complete description of the design of the sampled data control loops is given in part 2 of this thesis, with the software routines presented in Chapter 4.

2.4.4 Monitoring of the system status.

In-flight monitoring is used to provide a form of record, and to give an indication of the health (calibration and continued functioning) of the system. Information implying catastrophic failure of the processor is used to select the analogue back-up, which is discussed in the next section. Loss of the telecommand link will engage one of the automatic failsafe procedures. The remaining error data is transmitted to the ground.

The types of information telemetred are as follows:

A) All of the sensor measurements available to the control loops.

B) Control surface demands.

C) System status flags derived from the software, such as whether the system has initiated a software failsafe, or is above demanded height, etc. These flags concern the status of the software routines and the calibration of the circuitry within the digital control unit.

D) System status flags derived from the hardware, such as active analogue failsafe, state of charge of the battery stacks, etc. These flags indicate the condition of the system as a whole, and, unlike the above, will continue to be meaningful in the event of a processor failure.

As the telemetry operates independently of the System cycle, this information has to be available throughout the period and so must be stored in hardware interface latches. Furthermore, a certain amount of signal conversion has to be carried out for some functions, to make them compatible with the inputs to the telemetry encoder unit. This includes D/A conversion for the software generated analogue functions, and voltage level shifting for the bipolar sensor outputs. The hardware and software associated with these is given in sections 3.2.3.7 and 4.4.3.3

2.4.5 Failure detection and the back-up system

In larger systems, usually where life is at stake, the approach to this problem is to introduce redundancy in the flight control system. In order to reduce the probability of a complete failure to an acceptable level (often of the order of 1 in 10 million flying hours), triplex or quadruplex systems are called for, each with their own supplies, sensors etc.. These are connected in a majority voting scheme so that, if one or more systems give erroneous results, they are ignored. Due to the size and mission of the Stabileye aircraft, this was not sensible here. The cost and weight of a triple or quadruple redundant system would make such a level of security prohibitive for a mini R.P.V. Evidently however, some degree of

error detection and correction is required.

The types of errors to be catered for are, loss of command link, which results in a failsafe procedure, and failure of the processor system. This may be caused by conditions such as environmental noise, device failure, software errors, power loss etc.. Though some of these might only degrade the performance of an analogue autopilot, they are all likely to prove catastrophic in a digital system. Careful design of the software seeks to eliminate the occurrence of software errors, but the others require active detection and, if possible,, corrective measures. A software latch-up as a result of noise or power supply interruption is usually cured by resetting the processor, but more permanent failures require the use of a back-up system.

This is achieved by relinquishing command of the actuators to a basic analogue controller provided by the receiver unit. This gives direct pilot command and a rudimentary failsafe routine which will continue to function as long as power supplies to the receiver and servo-actuators are operational. The back-up system can be selected by the pilot via the command word, or automatically in the event that the processor malfunctions. The digital and analogue systems are interfaced in the controller selection unit which operates on the principal of two tri-state buffers driving the actuators, only one of which can be enabled at one time. If the analogue system is engaged, the digital controller can still be monitored via the telemetry. Though this provides redundancy of a sort, it does not incur the cost and weight penalties of a true duplex scheme. However it does not provide the same degree of protection as that provided by complete duplication.

Battery failures could still prove fatal if used by the back-up as well. Duplication of the battery supplies is discounted on the grounds of weight, so each supply is composed of a main stack capable of 2 hours duration and a reserve with a capacity for 15 minutes. These are so connected that a cell failure in one will not harm the other. Providing duplicate sensors is unrealistic due to their cost, so a failure, which might manifest itself via the telemetry, requires that the autopilot be dis-engaged. Servo failure can only be overcome by duplication and some form of clutch mechanism. This is not realistic here.

An error in the processor system will result in a disturbance of some form. Temporary problems, such as corruption of the programmable timer data, or the Servo cycle counter, will be restored by the software itself, as this re-loads all parameters and pointers at the start of the System cycle. A disturbance resulting in latched-up software (namely stopped or looping upon itself), or errors due to hardware failures will not do this. Hence an independent method of checking that the processor is still running is required.

This is achieved using a watch-dog timer. At the start of the System cycle the processor accesses a particular memory location. This triggers a monostable which has an output pulse duration of 50ms, or twice the System cycle period. If this device is re-triggered before the output pulse duration has lapsed, it will begin another 50ms period. When the processor is functioning correctly it will set the monostable every 25ms, so that the output, designated "Dig Fail Flag" (see Appendix A) will remain set continuously. If the system fails, "Dig Fail Flag" will change state, thus initiating a processor reset and automatically engaging the

back-up controller.

Resetting causes an un-maskable interrupt which re-starts the software and re-loads all of the counts, indexes, etc.. As corruption may have occurred, the control surface demands are set to midway and the command word defaults to no parachute, no airbag, autopilots dis-engaged and the back-up selected. This re-synchronises the software to the hardware and should clear any software errors. As the programme re-commences, "Dig Fail Flag" is set, thus engaging the digital system once more. The entire action occurs within 100us but, should the error persist beyond this, the reset signal, which is generated by the oscillator circuit described in section 3.2.2, is repeated. The default command word will operate until new information is received from the ground station.

If the problem is due to a c.p.u. card failure the software will not run at all, in which case resetting will have no effect and the back-up assumes control permenantly.

Chapter 3 Design of the digital flight control system electronics.

3.1 Overall review of the electronics.

The flight control system would normally be located in the electronics bay of the aircraft (see Fig. 1), along with the instruments and the battery pack. However, as this system is a prototype, and will not be flown with a payload, it was decided to site the instruments in their normal locations, but move the digital control unit and batteries to the payload bay and nose compartment respectively. This was to provide a better weight distribution and to take advantage of the greater size of the payload bay to allow easier access. A view of the digital flight control system prior to installation is shown in Fig. 7.

The main component is the digital control unit, which contains the c.p.u. card and all of the circuitry necessary to interface to the aircraft system. This is connected by a wiring loom to the other equipment, namely the instruments, the power supplies, the telecommand and telemetry systems, the controller selection unit and the actuators. The interconnection of these is shown in Fig. 8. The sub-systems will be described in the following sections with the exception of the telecommand and telemetry, which are described elsewhere (3,4). When referring to the circuit diagrams, the labels I.C 1, I.C.2, etc. correspond to the list of major components given in Table 1.

3.2 The digital control unit.

The main components of this unit are the c.p.u. card, a commercial

item which will be described in section 3.2.2 and the aircraft interface card, full details of which will be given in section 3.2.3. These are mounted within a custom built aluminium case (which also houses the voltage regulator circuits). Interconnections between these two boards are made via a backplane bus connected to the external systems.

The principal constaints involved in the design of the digital control unit are that size and weight should be kept to a minimum. Ultimately the size is dictated by the dimensions of the c.p.u. card, which was one of the smallest available at the start of the project. The low weight requirement gives rise to a restriction of the power consumption, as the heaviest component in the system is the battery box. Hence low power CMOS devices are used wherever possible.

3.2.1 The choice of the microprocessor.

While a CMOS processor such as the RCA 1802 might have been attractive, there are other factors which lead to the adoption of the Texas Instruments 9900 (5). To allow sufficient accuracy of internal calculations without recourse to double precision arithmetic, a full 16 bit machine is called for. This rules out many of the first generation processors, such as the Zilog Z80 and the Motorola 6800. Furthermore, at the initial stages of the project, the TMS 9900 was available at military specification, the RCA 1802 was not. Possibly the most important consideration was the interrupt structure. A processor employed in a stand-alone control application is likely to have to accommodate a number of separate interrupts, and so it should be flexible and easy to use in this respect. The TMS 9900 uses memory-to-memory architecture, with only the minimum of internal

registers, so that interrupt initiated context switching is particularly simple. When used in conjunction with a TMS 9901 peripheral device (6) it can accept 16 separate , prioritised interrupts, each one capable of being enabled, or disabled using single I/O instructions. For these reasons, the TMS 9900 was the most suitable processor available at the hardware definition stage of the project.

3.2.2 Specification of the c.p.u. card.

The option to produce a custom c.p.u. card was not taken because it was unlikely to have been significantly different from the chosen commercial item. This is a Texas Instruments TM 990-100M (7). A block diagram of this card is given in Fig. 9 and a description of the relevant details follows:

a) The board has 256x16 bits of random access memory (RAM), expandable to 512x16 bits. The RAM is used for all register locations for calculations, counts, indexes etc. During software development the full 512 words are used for programme storage as well, but no more than 256 words are required in flight.

b) It has 1024x16 bits of eraseable, programmable, read only memory (EPROM), expandable to 2048 or 4192 words. This is used for programme storage due to its non-volatile nature. The on-board monitor programme, required during software development, occupies 1024 words.

c) All data, address and control lines are accessible for system expansion.

d) RS232 or 20mA current loop interfaces are available to allow connection to a terminal.

e) There are two programmable timers, one of which is used to provide

the timing signal to synchronise the servomotor driving and sensor sampling operations.

f) It supports the TIBUG interactive monitor programme which provides a comprehensive range of software development facilities.

3.2.2.1 Modifications to the c.p.u. card.

Additional circuitry has been added to the c.p.u. card primarily to provide an automatic resetting facility. These modifications are shown in Figs. 10a and 10b. The I/O buffer circuit is used to buffer I/O ports P0 to P5 from the TMS 9901, which provide the control signals for the pulsed functions. These are; the camera (if fitted), ignition cut, parachute deploy, parachute jettison (if fitted) and airbag deploy systems. Apart from buffering these lines, this circuit also gives logic inversion. This is necessary to prevent accidental firing of the actuators during a reset, or power-down situation, when the I/O ports assume an indeterminate high impedance state. In this condition the pull-up resistors shown in Fig. 10a tie the inputs of the CD4049 inverter buffers (I.C.1) HI so that the outputs are all LO, which will not trigger the ignition cut relay or the explosive actuators for the parachute and airbag deployment.

Fig. 10b shows the watchdog timer and automatic reset circuit. The N755 timer device (I.C.2) is configured as a self-starting astable multivibrator which produces a 5kHz square wave with a mark-space ratio of approximately 10:1. When applied to the "$\overline{\text{PRES.B}}$" input of the c.p.u. card (7), this initiates a processor reset when in the LO state. The software then has approximately 200us to disable the automatic reset procedure by setting the output of the watchdog timer monostable before the next occurrence of a reset request. The

CD4047 monostable (I.C.3) is retriggerable with a pulse duration of 50ms. During correct operation this is set every 25ms via the "$\overline{\text{I/OSEL1}}$" address, and so remains in the Q = HI state, thus gating out the reset pulses. Alternatively, the automatic reset facility can be disabled in favour of a manual reset switch on the c.p.u. card, for use during software development.

The software generated system status flags are provided by I/O ports P8 to P15. These are not buffered in the same way as the actuator driving signals as they would not compromise the integrity of the system when left floating during a reset. A list of I/O bit usage is given in Table 2.

3.2.3 The aircraft interface card.

3.2.3.1 Physical description.

This is a double eurocard carrying all the circuitry necessary to interface the c.p.u. card to the aircraft system. Interconnections are via two 64-way edge connectors to the backplane, and a 20-way ribbon connector for the height-zero store switches. A block diagram of the board is given in Fig. 11 showing the functions performed. Each part will now be described in terms of its construction and operation.

3.2.3.2 Address decoding.

A circuit diagram for the address decoding circuit is given in Fig. 12. Most data transfer to and from the c.p.u. is achieved by treating the external circuitry as a series of locations in the processor

memory map. Tri-state buffers are used to interface to the data bus, controlled by decoded address signals. Hence the processor simply reads from, or writes to the required location. The address decoding circuit provides 16 lines corresponding to the memory locations °C3E0° to °C3FE°. When accessed, each line produces a LO going pulse of about 500ns duration.

As this is working at full processor clock speed (3MHz), low power Schottky TTL devices are used instead of CMOS, so pull-ups are required when the outputs are used to drive CMOS inputs (8). Timing diagrams for memory read and memory write cycles for the TMS 9900 are given in Fig. 13.

3.2.3.3 Eight channel servomotor driving circuit.

Control surface position is specified in the microprocessor system as a number, so a synchronous counter is used to convert this into a pulse duration. The servomotors require input pulse-widths between 0.9 and 1.9ms to give their full range of movement. These pulses are composed of a fixed part of 0.9ms, and a variable part ranging from 0.0 to 1.0ms duration. The fixed part is hard-wired in the counter output gating which detects when the count corresponds to 0.9ms after the zero count. The variable part is generated by loading a positive number into the counter stages, and then counting down through zero. The sign bit is achieved with the most significant bit of the counter, which uses the convention HI = positive, and LO = negative.

The circuit diagram is shown in Fig. 14. CMOS is used throughout, except for the input buffers (I.C.13,14) which are 74C374's. These belong to a logic family which uses CMOS cell construction techniques, but are capable of driving one low power

Schottky TTL input without modification. The counter (I.C15,16,17) runs off a 1:1 mark-space ratio, 1.5MHz square wave, derived from the 3MHz processor clock. This dictates the number of bits required by the counter. The maximum value for the variable part of the servomotor drive pulse is 1ms, or approximately 1515 counter clock periods. Expressed in binary, this requires 11 bits, with one extra for the sign, making 12 in all. (This means the quantisation effect inherent in the servomotor driver signals is much smaller than the expected mechanical errors due to stiction and linkage backlash.)

The output from the counter gating, a CD4078 (I.C18), eight input NOR, is connected to one of eight servomotor lines using a CD4724 (I.C.20) addressable latch.

Operation of the circuit is as follows. On receiving the servo interrupt signal, the processor writes a number to the input latches located at °C3F0°, composed of address and data bits. The 11 least significant bits (D15 to D5) are loaded assynchronously into the counter, along with a sign bit which is hardwired to a logic HI. The 3 address bits (D3,D2,D1) select the required servomotor channel. Gating at the output goes LO setting the selected servomotor driving line to a HI, and releasing the clock enable. The counter decrements (through zero) until the state corresponding to -0.9ms is reached, (contents = 0010 1010 1100) whereupon the gate output goes HI again, clearing the selected servomotor channel and disabling the counter clock. The circuit remains in this state until next accessed by the processor. Finally, the output of channel 7 is used to generate the servo check interrupt signal, which confirms calibration of the counter.

3.2.3.4 Sensor input buffer circuits.

As stated previously, the sensor outputs and other voltage levels are measured using a 12 bit A/D converter in conjunction with an analogue multiplexor. The software for the control loops uses fixed point arithmetic, so some scaling standard has to be chosen for each loop. This must be a compromise between the possibility of overflow and loss of precision due to quantisation (23,25). In general the maximum value is taken to be about twice the largest possible demand. Hence the sensor output voltages are amplified by operational amplifier buffers to match this chosen limit to the A/D converter input range. A circuit diagram of these buffers is given in Fig. 15.

Roll information is derived from one axis of a dual axis attitude gyroscope configured to provide +/-5.0V output at +/-1.396rad of roll. The maximum demanded roll attitude is +/-1.047rad but this would not be encountered in normal use. Hence the roll loop is scaled to +/-1.396rad maximum, so that a simple unity gain voltage follower is all that is required here. Feedback resistor R1 matches the impedance of the potentiometer at the gyroscope pick-off, and R2 is included to tie the input to ground in the event that it is left open circuit. This prevents the indeterminate operational amplifier output damaging the analogue multiplexor.

Pitch information is derived from the other axis of the gyroscope, which is arranged to give +/-12.0V for +/-0.698rad attitude. The demand range of the pitch channel is +/-0.140rad so the software is scaled to a maximum of +/-0.291rad. This allowed the use of a unity gain voltage follower with input protection as before. The

zener diodes ZN1 and ZN2 are to limit the voltage seen by the analogue multiplexor to the range +/-5.0V. If the pitch attitude exceeds +/-0.291rad the output of the operational amplifier buffer will attempt to exceed this range, the voltage difference is then dropped across the output resistor of the operational amplifier, (250 ohms for the TL084 used).

Yaw rate is measured using a rate gyroscope giving 3.438mV/rad/s, with a maximum output of +/-6.0V. The yaw rate is expected to be in the range +/-0.140rad/s. By employing an operational amplifier buffer with a gain of 4.167 the voltage presented to the analogue multiplexor, and thus used to scale the software, is calibrated to give +/-5.0V for yaw rate = +/-0.349rad/s. Resistors R3 and R4 give this value of gain, while resistor R5 and diodes ZN3 and ZN4 are included for the same reasons as those in the pitch buffer.

The aircraft height is deduced by measurements taken from a barometric pressure sensor, which is an absolute sensor and must be referenced to the pressure at the ground. Unlike the other sensors, this device is permanently installed, so no provision need be made for the open circuit case. The expected range of sensor voltages during flight conditions is subject to:

a) The pressure/voltage characteristics of the transducer, (a National Semiconductor LS1601A, see Appendix B).

b) Variation of ambient sea level pressure, (980.0 to 1040.0millibars is accommodated).

c) Pressure/altitude gradient of a standard atmosphere.

d) The height above sea level of the ground station, (0.0 to 300.0m allowed).

e) The height of the aircraft above the ground station, (0.0 to

1000.0m allowed).

The highest pressure will occur when the aircraft is at sea level with maximum ambient pressure, giving 10.7V from the transducer used. The lowest pressure will occur with the aircraft at 1300.0m above sea level with the minimum ambient value. This gives a voltage of 6.1V with this device.

The height input buffer is required to convert this to a form compatible with the A/D converter, hence an offset must be included. This is achieved using the circuit shown in Fig. 15. Resistors R6, R7 and R8 are dictated by the relationship:

$$V(out) = -\left(\frac{R8}{R6} V(transducer) + \frac{R8}{R7} V(offset) \right)$$

where

$$V(offset) = -12.0V.$$

R9 is the parallel combination of these to minimise input offset effects. Hence, the input/output relationship is arranged so that for V(transducer) = 6.1V, V(out) = 4.58V, and for V(transducer) =10.7V, V(out) = -4.62V. This satisfies the range requirements for the analogue multiplexor and defines the scaling so that a height difference of 1000.0m, under given ambient conditions, gives a voltage difference of 5.0V.

The airspeed indicator includes its own offset and calibration circuit, and is arranged to provide a voltage output between 0.5V and 4.5V for airspeeds approximately equal to 20 and 50m/s. Precise details are given in Appendix B. Hence, all that is required here is a voltage follower with input and output protection as before.

Finally the calibration of the A/D converter is checked by measuring a fixed voltage reference of 4.3V, provided by a zener diode voltage reference circuit.

3.2.3.5 Analogue multiplexor and A/D converter circuit.

The analogue multiplexor and A/D converter (9) control circuit is shown in Fig. 16. As configured, the analogue input range of this device is +/-5.0V, while the digital output is in 12 bit, complementary, two's complement format. Conversion is started by a "CONVERT COMMAND" signal, and takes approximately 25us, whereupon a flag, referred to as "STATUS", is set, and the output data becomes available. The analogue multiplexor device (I.C.25), an RCA 4051, directs one of eight channels to its output according to the state of three select inputs or, alternatively, can be set to a high impedance state using an inhibit input.

The operation of the circuit is as follows. The channel number and inhibit flag (inactive) are loaded into the address latch (I.C.24) by writing to the 4 most significant bits of memory location °C3F4°. This sets up the control inputs for the analogue multiplexor and connects the required input channel to the A/D converter. Writing to °C3F6° then generates the "CONVERT COMMAND" signal which starts the A/D conversion. On completion the "STATUS" flag goes HI, loading the 12 bit output from the A/D converter into two octal latches (I.C.26,27). The processor, which has been delayed for 30us after initiating the conversion, reads the contents of these latches by accessing memory location °C3E2°.

3.2.3.6 The telecommand port.

The circuit diagram for the telecommand port, which interfaces the

receiver to the microprocessor system is shown in Fig. 17. When new telecommand data is available it is presented at the output of the receiver for approximately 7us, along with a flag called "NEW DATA READY", which goes HI for this period. This flag is delayed for lus to allow the outputs of the receiver to settle, and then used to load the address and data words into two octal buffers (I.C.30,31), at the same time setting a d-type latch (I.C.32) which generates the "$\overline{\text{INT4}}$" interrupt. This signals to the processor that there is new data available which will be read when the interrupt mask is set to the appropriate level. The processor accesses the port by reading the contents of memory location °C3E0°, which also clears the interrupt generating d-type latch.

3.2.3.7 Telemetry data conditioning.

Analogue inputs to the telemetry encoder must be in the range 0.0V to +5.0V. This is suitable for some functions (such as "$\overline{\text{RESET}}$"), but bipolar signals from the sensors require an offset and a gain adjusting circuit as shown in Fig. 18. The gains, and hence the resistors are chosen to match each signal to the correct range.

Due to the predominance of analogue telemetry channels, and also to reduce the number of lines in the aircraft wiring loom, the servomotor demands and height information functions, which are generated in the software, are presented to the telemetry encoder as analogue signals. This requires digital to analogue (D/A) conversion as shown in Fig. 19. A total of six AD7524, eight bit latchable D/A converters (I.C.34-39) with the necessary addressing and output buffering are used. These are arranged as three 16 bit words, loaded using memory-write operations. Throttle demand (THR DEM) and aileron

demand (AIL DEM) occupy the lower and upper bytes of location °C3F2°. Absolute height (HT ABS) and height error (HT ERR) are located in the lower and upper bytes of location °C3F8° and rudder demand (RUD DEM) and elevator demand (EL DEM) are located in the lower and upper bytes of location °C3FE°.

The digital telemetry information does not need any form of conditioning. The software based system status flags are generated in the microprocessor's I/O field on the c.p.u. card, and the hardware based flags are simple logic outputs.

3.2.4 The voltage regulator circuits.

The electronics and instrumentation are all supplied with voltage rails derived from voltage regulators (10) located within the digital control unit. The circuit diagram is given in Fig. 20. All regulators and smoothing capacitors are mounted on one board on the side panel, with the exception of the +5.0V regulator. To aid heat dissipation this is mounted on, though electrically isolated from, the aluminium panel forming the base of the unit. The regulators are rated to have at least twice the current handling capacity of the expected requirements, and the unregulated input voltage should be at least 2.5V greater than the nominal output voltage for correct operation.

The +/-5.0V and +/-12.0V rails are provided by fixed output devices, for which no external components other than smoothing capacitors are needed. The +/-6.0V rails, which supply the rate gyroscope and the analogue multiplexor chip, are provided by variable voltage regulators. These require two resistors each (R1, R2, R1a, R2a) to set up the output voltage level, given by:

$$V(out) = 1.25V(1.0 + \frac{R2}{R1})$$

The current rating of the regulators, along with a breakdown of the supply requirements for each voltage rail are given in Table 2.

3.3 The height-zero storage.

The barometric pressure reading corresponding to the zero height has to be programmed by the operator prior to flight. As this information is used in the failsafe routine it must be stored in a non-volatile medium, hence a bank of switches , mounted on the fuselage side are used. The circuitry involved in the height-zero store is shown in Fig. 21. At some stage during the pre-flight procedure the operator depresses the "HEIGHT ZERO SET" push button, signalling the software to write the 8 most significant bits of the pressure sensor reading to a bank of l.e.d's, also situated on the fuselage side. The operator then sets the switches according to which bits are illuminated.

When not used for this purpose the l.e.d's can display a combination of hardware and software based, aircraft status flags for checkout use. In order to reduce power consumption the display can be disabled prior to launch.

A 74LS240 octal inverting buffer (I.C.32) is used to drive the l.e.d's. The inputs to this are either provided by the I/O ports P8 to P15 on the c.p.u. card, or in the case of the three most significant bits, by hardware outside the digital control unit. These three inputs are multiplexed using a 74LS157, quad 2-to-1 line device (I.C.43). This multiplexor is controlled by the "HEIGHT ZERO SET" button, which is also used to drive I/O port P7 to select the height-zero set software.

The switches drive the inputs of an octal tri-state buffer (I.C.44), the contents of which are accessed by reading memory location °C3E2°. This action also loads the buffer with the switch data, and so the operation must be performed at least twice when altering the height reference. In practice this does not pose a problem as the switches are read many times per second while the "HEIGHT ZERO SET" button is held down.

3.4 The controller selection and actuator driving unit.

This unit serves the dual purpose of interfacing the digital and analogue control systems, and providing the driving circuits for the explosive actuators which operate the parachute and airbag deployment. The circuit diagrams for both functions are shown in Fig. 22. The two control systems are interfaced using 74C374 octal tri-state buffers (I.C.47,48). Hence the actuator inputs are treated as a data bus, with only one of the two sources having access at any one time. The selection of the control system takes into account the continued functioning of the digital system, and the relevant bits in the command word. This is accomplished using the descrete logic shown in Fig. 22, which also provides buffered versions of "DIG/AN SELECT", "DIG FAIL FLAG" and "SQUELCH" for telemetry purposes.

The parachute and airbag deployment systems use explosive actuators to release mechanical catches. These actuators require electrical pulses of 1.0s duration and will draw approximately 0.5A during detonation. The timing of the pulses is provided by software means, or, in the case of the analogue back-up system, a square wave of 5.0s period, derived from circuitry within the receiver, is used. These two systems are interfaced in the same way as the servomotor

driving signals. The relevant outputs from the tri-state buffers are used to drive the base inputs of LM295 ultra reliable power transistors (I.C.49,50,51). These devices have integral base input resistors which enable them to be driven directly off CMOS or TTL logic levels. They also possess output current control so that they can sink the relatively high, but short duration, current level required during firing, without sustaining damage.

The explosive actuators are connected between the servomotor supply rail and the transistor collectors so, when energised, they see a low impedance path to ground. They can be disarmed using a switch which removes them from circuit, and connects an l.e.d. and resistor in their place. This facilitates testing without detonating an actuator. An "ARMED" status flag is provided for telemetry purposes. Note that the actuator arming switch, testing indicators, and "ARMED" status flag are all located in the battery supply control unit, which will be discussed in section 3.5.2, and is shown in Fig. 25. The zener diode in the "ARMED" status flag circuit is included to limit the level of the telemetry input to 5.0V, as the fully charged level of the servomotor battery supply is 7.2V. They have been described here for continuity.

The ignition cut actuator is a relay as shown in Fig. 23. This is mounted in the engine compartment ,but the control signal, "IGCUT(OUT)" is derived from the controller selection unit. It is desirable that the ignition is not cut in the event that the digital system fails, so the "IGCUT(OUT)" line is tied to ground when operating under the analogue back-up.

3.5 The power supplies.

All electrical power for the DFCS is supplied by nickel-cadmium rechargeable cells. As described previously in section 3.2.4, the electronics and instrumentation are powered by regulated voltage rails. These are all supplied with a main and reserve source to reduce the risk of power failure. The battery system is divided into two parts, the cells themselves, located in the nose where they are easily removed for charging, and the control unit which is situated in the mid compartment. Conventional noise minimisation techniques, such as extensive use of decoupling capacitors and separate supply lines using heavy guage wire are used throughout.

3.5.1 The battery box.

The battery stacks are shown in Fig. 24. As can be seen there are four different sections, namely an 8.4V supply, two 15.6V supplies and a 7.2V supply. These voltages refer to nominal values assuming one nickel-cadmium cell produces 1.2V, but in practice, fully charged, no-load voltages are about 10% higher than this. These sections provide the +8.4V, the +15.6V and the -15.6V supplies for the regulator circuits, and the +7.2V supply for the servomotors. The "GND" terminals are not connected at this point so the cells can be re-charged from a single polarity source.

A section supplying a regulator circuit has to be at least 2.75V in excess of the regulated output voltage. This is accounted for by a forward biassed Schottky diode connected in series with the stacks (approximately 0.25V), and the regulators, which require a margin of 2.5V to ensure correct operation.The +8.4V section supplies the +5.0V regulated rail for the DFCS. This provides approximately 1.0A in flight configuration, (the amount of RAM installed in the c.p.u. card

effects this, only 256 words are used in flight), so the main stack , which is designed to operate for two hours, has 2.0Ah capacity. The reserve, which is intended to provide 15 minutes running, to allow a software failsafe to take place, consists of two, 110mAh, 8.4V batteries connected in parallel.

Similarly the +15.6V rail, which gives +12.0V regulated, supplies 0.35A for the electronics and 0.5A for the attitude gyroscope, and so has a capacity of 2.0Ah. As before, the reserve is composed of two, 110mAh, 16.8V stacks connected in parallel. The -15.6V rail (-12.0V regulated) provides approximately 0.15A and so main and reserve capacities of 0.45Ah and 0.1Ah respectively, are used.

The +7.2V section, which is used to supply the servomotors and explosive actuators without regulation, is composed of two identical stacks, each capable of maintaining the supply for 2.0 hours. These have a capacity of 1.2Ah each.

The interconnection, monitoring and control of these stacks is provided by the battery supply control unit.

3.5.2 The battery supply control unit.

This unit provides for the control and distribution of the battery supplies throughout the system, including the DFCS-on/off control and the interfacing of the main and reserve stacks. The circuit diagram is as shown in Fig. 25. The unit is mounted so that the switches and the test indicators, described in section 3.4, are accessible. The purpose of the battery reserve changeover circuit, as shown in Fig. 26, is to check the voltages of the main stacks , and engage the reserves if they fall below a given level. When any one of the

reserves is engaged a telemetry flag, referred to as "BATOK", is cleared, informing the operator that there is only 15 minutes of electrical power remaining.

The output of a nickel-cadmium cell supplying a constant current is as shown in Fig. 27. As can be seen the voltage remains reasonably constant, until such time as a threshold level is reached, whereupon it falls off rapidly. This value corresponds to the nominal voltage of the stack (see 3.5.1), and is the reference level used for comparison purposes. The operation of the +15.6V section of the circuit is as follows:

The output of the main stack is reduced via a resistor divider, which is arranged to give +5.0V at the changeover voltage. This is checked against a +5.0V reference, obtained from a precision regulated source, using an LM339 voltage comparator (I.C.52). Positive feedback is provided using a 1.0 mega-ohm resistor to give a certain amount of hysteresis, and so prevent ringing at the changeover condition. The output voltage excursion of the operational amplifier is modified using the resistor and zener diode combination shown, to be compatible with a 5.0V CMOS logic input. This is to buffer the signal, before driving a discrete transistor base input, and to allow the use of combinational logic to generate the "BATOK" flag. At the changeover point, a BC184C, transistor is used to close the contacts of a miniature relay and connect the +15.6V reserve stack in circuit. The main stack is still connected, so there is no power supply drop-out as the relay changes over. The diodes in series with the battery stacks are included to prevent a fault in one interfering with the output of the other. Schottky diodes are used as these have a lower forward bias voltage drop. The operation of the +8.4V, and the -15.6V comparator circuits is similar to the above.

The DFCS-on/off switch is used to disconnect all power from the aircraft, with the exception of the battery reserve changeover circuit, which must be supplied with power rails whenever the battery box is installed. This prevents the possibility of the sense inputs from the main stacks, which are always present when the battery box is connected, being energised while the supplies to the comparators are not. The telemetry switch is used to disable the transmitter until required. This is because the aircraft is usually checked prior to launch by observing the output of the telemetry encoder directly.

A duration test was carried out on the battery supplies with all the aircraft systems connected. This was achieved by fully charging the battery stacks, then observing the main, reserve and aircraft voltage rails, at 5 minute intervals, while continuously excersising the controls. The objective of two hours duration was realised, as the changeover took place after 120 minutes. The digital system continued to operate for a further 30 minutes, whereupon the +8.4V rail failed completely. The analogue back-up, which does not use this supply, was still operational at this stage.

4.0 The design of the system software.

4.1 Description of the software development technique.

The software is written in assembler code, a process which takes longer than using a high level language, but results in more economical use of the EPROM storage. The editting and assembly stages were performed on a PDP-11/34 minicomputer (11) to provide the machine code and a listing. This code was then installed in the memory space of the c.p.u. card and checked under the control of the TIBUG monitor programme (12). An example of the listing is provided in Appendix C.

4.2 Brief review of the TMS 9900 microprocessor.

The reasons for choosing the TMS 9900 were given in section 3.2.1. Its main feature, from a software standpoint, is its use of memory-to-memory architecture, so that it does not appear to have accumulators. Instead it specifies a workspace, which is a file of 16 words, anywhere in the RAM. These are referred to as registers (0-15), and can all be used for arithmetic and bit manipulation purposes. Each particular activity can have its own workspace, making context switches (e.g. interrupts) a simple process. This is controlled by three internal hardware registers; the programme counter, which indicates the location of the next instruction, the status register, which is set according to the result of the previous operation, and the workspace pointer, which contains the address of the current workspace. The interrupt context switch causes the programme counter and the workspace pointer to be replaced by new

values from memory, and the old internal registers to be stored in R13, R14 and R15 of the new workspace. Hence interrupts can be nested, provided they do not share the same workspace.

Each interrupt has a priority level, 0 being the highest and 15, the lowest. If the incoming interrupt has a higher priority than the prevailing condition, as indicated by a mask in the status register, it is acknowledged, otherwise no action takes place. This mask can be loaded directly by software instruction, and is automatically set to the next highest priority on acceptance of an interrupt.

4.3 The structure of the system software.

4.3.1 Timing and synchronisation.

The software is synchronised through interrupts. One master clock is used, with all activities, excepting telecommand reading, performed relative to this. The timing of the System and Servo cycles has been described previously in section 2.4.1. These are generated using interrupt level 3, referred to as the servo interrupt and derived from the programmable timer. Two of the remaining interrupts, namely, telecommand (level 4) and servo check (level 6) are only enabled at given periods during the System cycle. This is to prevent the possibility of corruption due to nesting, as they share the same workspace. The overall timing diagram of the software is shown in Fig. 28.

4.3.2 Structure.

The software is arranged on a foreground/background basis. The foreground operations are all time-critical, such as servomotor driving and sensor reading. These are initiated by the highest priority interrupt in general use, so that they are always performed immediately. The background software deals with the functions that are not synchronised to the programmable clock, such as telecommand reading, telemetry and house-keeping activities. These are performed once per System cycle, during the periods when the foreground routines do not require the processor.

Fig. 29 gives the overall flow diagram of the foreground software. The servo interrupt routine forms the basis of this structure by directing the programme flow to the required section, according to the value of the Servo cycle counter. The calculation of each new servomotor position by the relevant control law, is performed in the period immediately prior to its own Servo cycle. This minimises the delay between sensor reading and servomotor update for a particular loop. On completion, control is returned to the next location in the background software, using the stored programme counter. During the eighth Servo cycle the return vectors are modified to point to the start of the background software once again. The role of the servo interrupt routine in the software structure is described in more detail in section 4.4.4.1.

The background is not structured, with the routines being performed in the order shown in Fig. 30.

4.3.3 Workspaces and the transfer of information.

Most sections have their own dedicated workspace, which is used exclusively for a particular activity. Some means of transferring data between workspaces is required, and this is achieved by the use of symbolic (or label) addressing. That is, certain workspace registers are given labels which allow them to be accessed from any part of the software. A complete list of the functions and labels of each workspace register is given in Appendix D.

4.4 Description of the routines.

4.4.1 Types of routine.

There are three types of routines; those in the foreground, which calculate the next servomotor values, the background, which performs all of the non-time-critical system activities, and the utility programmes which can be called as subroutines from anywhere in the software.

4.4.2 Utility subroutines.

These can be called from any part of the software using the branch-and-link instruction, which stores the return address in R11, before jumping to the start of the subroutine. These use the same workspace as the calling routine, so input data can be set up in registers before use.

4.4.2.1 Multiply routine (MULT).

A discussion of the number system used, and the treatment of multiplication is given in Chapter 6. The multiply instruction for the c.p.u. multiplies two unsigned, 16 bit numbers to give a 32 bit result. This is not compatible with the fixed point, two's complement number system employed in the control loops. The multiplicand in R8 is multiplied by the coefficient, or multiplier, stored at a location pointed to by R12. This latter part is a 16 bit positive binary fraction, with the binary point to the left of the most significant bit. This means that the most significant word of the 32 bit result will be specified in the same fixed point standard as was the multiplicand. The routine will also carry out a left shift, N times, where N is specified in R10.

There is no provision for a sign bit in the multiplier, so negative coefficients must be accommodated by negation of the result in the calling programme. The routine will cater for a negative multiplicand, by actively negating R8 before and after the multiply instruction, if necessary. Overflow, arising from the left-shifting process is detected, whereupon the result is replaced by the maximum positive, or negative value allowed. The most significant word of the result is returned in R8, having first been rounded to 16 bits according to the state of the least significant word at R9. The coefficient pointer in R12 is incremented to point to the next location if required.

The flow diagram is shown in Fig. 31. As there is no simple means of shifting the 32 bit result as a single word, this process is performed first, provided it does not lead to overflow. The multiplicand in R8 is stored temporarily in R9, then R10 is checked

to see if shifting is required. If so, R9 is shifted once and checked for overflow. If no overflow occurs, R8 is shifted as well, and R10, the shift count, is decremented. This is repeated until R10 is zero, or until overflow does occur, whereupon R8 and R9 are left as they were before the shift that caused the overflow in R9. The negative data flag, R13, is set if necessary, then the multiplication of the absolute contents of R8, and the coefficient pointed to by R12, is carried out. The result is negated if the negative flag is set and, if the shift count in R10 is zero, control returns to the location pointed to by R11. If R10 is not zero, it is possible that the result can now be shifted without overflow, if the product of the multiplier and the multiplicand is less than 0.5. The remaining shifts are performed, using saturation in the event that the overflow persists, prior to returning via R11.

4.4.2.2 Sensor reading routine (SENS).

This includes the ability to read the specified channel several times, accumulating the result to reduce the effect of high frequency noise at the analogue input. The A/D converter device produces a 12 bit, complementary, two's complement number, which must be converted to the 16 bit standard of the software. The flow diagram is given in Fig. 32.

The accumulator, R12, is cleared, then the channel required, which is specified in the four most significant bits of R10, is set-up, and a short delay is observed to allow the channel to settle. Conversion is started, and a further delay of about 30us occurs, achieved by a temporary count in R9, before the result is read into R9. This is inverted to give 12 bit , two's complement data, then

shifted four bit-places to the left, followed by three bit-places to the right, using arithmetic type shifts. These two operations have the effect of multiplying the number by two, and filling the four most significant bits of the result with the original sign bit. This value is added to the accumulator, then R8, which specifies the number of times the channel is to be read, is decremented. The process is repeated while R8 is greater than zero, then control returns to the address pointed to by R11. (Note that if R8 was set to zero before calling the routine, the specified channel is read once only).

4.4.2.3 Saturation adding routine (SATAD).

The flow diagram is given in Fig. 33. This adds the two's complement numbers in R7 and R8, and returns the result in R8, while R7 remains unchanged. If overflow occurs, the result is replaced by the maximum positive, or negative number allowed (°7FFF° or °8001°). If the difference between R7 and R8 is required, one of the registers is negated prior to calling SATAD. Control is returned via R11.

4.4.2.4 Pilot mode output scaling routine (SCALE).

The pilot commands are specified as 8 bit words with a range of °00° to °FF°, while the servomotor driver requires a number in the range °0000° to °05EB°, (see section 3.2.3.3). This multiplication is carried out using a shift and add routine as shown in Fig. 34. SCALE takes less time than using MULT, and only corrupts R13 and R14 of the calling workspace. The error due to the approximation (0.06% at full range) is not significant for the pilot mode.

4.4.3 The background software routines.

4.4.3.1 Telecommand interrupt enable.

The flow diagram is given in Fig. 35. The servo check routine, and the eighth cycle routine have re-directed the programme to the start point, where the interrupt mask is set to 3 and the processor is put into an idle state (see section 4.4.3.6). On receiving the next level 3 interrupt, the processor performs the servo routine, and then returns to the instruction following the idle. This is the start of the background software.

The telemetry workspace is loaded and the interrupt mask is set to 4, so that if a new command is available, the previously disabled telecommand interrupt routine will now be performed (see section 4.4.3.7). A counter, referred to as TCKNT (see Appendix D), is incremented to give a measure of the time since the last command, which, if in excess of 0.5s, will initiate the failsafe routine and set the software failsafe flag in the telemetry. Otherwise the programme moves directly to the end of the failsafe section where it re-triggers the "DIG FAIL FLAG" monostable, and procedes to the telemetry routine.

4.4.3.2 The software failsafe routine.

The failsafe procedure is as shown in Fig. 4, and has been described in section 2.3. The flow diagram is as shown in Fig. 36. On entering the failsafe condition, the command word is over-written to select autopilot and height lock only. Similarly the height and rudder

demands are set to the maximum, and middle position respectively. The timing counter in R5 is checked, and if less than 5.0s, which signifies phase 1 or phase 2, the throttle is set to maximum. The aircraft height is compared with a safe height of 100m, and if it is below this height, a bank angle of 0.174rad is loaded into the roll demand to perform the spiral climb of phase 1. This is repeated until a height of 100m is reached (phase 2), whereupon the roll demand is set to zero, and the timer counter is incremented. Phase 3 is initiated after 5.0s. This clears the throttle demand and activates the ignition cut, airbag and parachute bits in the command word. The 1.0s delay before deploying the parachute and airbag are provided in the oneshot routines, see section 4.4.3.4.

If at any time new telecommand information is received, the failsafe routine is omitted and TCKNT and the failsafe timer counter are cleared, thus resetting the system. Finally, the "DIG FAIL FLAG" monostable is set by writing to location °C3F6° irrespective of the state of the failsafe condition.

4.4.3.3 The telemetry preparation routine.

This section monitors the calibration of the A/D converter and then combines all of the telemetry information, (which is generated at various points throughout the software), into a series of bytes, before presenting them to the telemetry conditioning hardware. The flow diagram is shown in Fig. 37. The telemetry workspace is used, with TLPORT, the software generated flags, occupying the least significant byte of R4. The throttle and aileron demands occupy R2, and the elevator and rudder demands use R7.

TLPORT is dealt with first. The three most significant bits are

set according to the state of the autopilot, height lock and heading hold selects in the command word, and then the calibration of the A/D converter is checked. This is done by measuring a 4.3V reference voltage and ensuring the A/D converter gives a result within 1% of the correct value. The A/D error bit in TLPORT, is set if it exceeds this tolerance. At this stage the interrupt mask is set to 3 and the telecommand interrupt input is disabled to prevent the possibility of corruption later in the System cycle. The servomotor driver calibration check bit is then set depending on the state of a flag manipulated within its own interrupt handling routine. The last remaining bit is the overflow flag, which may have been set by the arithmetic procedures, MULT and SATAD, during the previous cycle, and does not need any handling here.

The servomotor demands have to be rescaled to the correct range, and are then stored in individual bytes at R2 and R7. Presenting the telemetry data is achieved by moving the contents of these registers to locations °C3F2° and °C3FE°, while TLPORT is loaded into I/O ports P7 to P15 using a multiple bit CRU instruction (5). This latter activity is omitted in the height-zero set mode. Finally, the overflow bit is cleared in preparation for the next cycle.

4.4.3.4 The oneshot routines.

The ignition cut, airbag and parachute deploy functions can only be performed once, and are irreversible, hence they are referred to as the oneshots. Demanding the parachute will also deploy the airbag and cut the ignition, the reverse, however, is not true. The pulses and voltage levels required to trigger the actuators are produced by manipulating bits in the I/O field (see Table 2). Timing is achieved

by counting System cycles. The flow diagram is shown in Fig. 38.

The software checks if the parachute deploy bit is active, and if so, increments a counter in R1. Further activity is not commenced until ten consecutive, active commands are received. If at any time the parachute deploy bit is inactive, R1 is cleared and the procedure is re-started. Once accepted, flags (R0, and bit 5 of R6) are set to inform the throttle, and ignition cut routines respectively. A delay of 1.0s is observed by waiting until the count in R1 reaches °32°, whereupon I/O bits °12° and °14° are cleared, thus triggering the explosive actuators. These bits are reset to ones when R1 has reached °5A°, giving 1.0s pulses, and signifying the completion of the parachute deployment routine. If this stage is reached, the remaining oneshot routines are omitted, as they will have been fired previously.

The other oneshot procedures are similar to the above, without the complication of actuating all three functions. Ignition cut counts ten active commands using R2, before clearing I/O bit °11° to energise a relay. Airbag deploy uses R3 as a timer counter, and generates firing pulses as in the parachute routine.

4.4.3.5 Height data preparation routine.

This section provides all the necessary operations for the height loop. These are, setting-up the height-zero, referring subsequent readings to this value to give the relative height, preparing HTERR and HTABS for telemetry, and finally performing the control law for the outer loop of the height lock. A flow diagram for this section is given in Fig. 39.

The height workspace is installed, and then the present output

from the barometric sensor is measured, and stored in R7. The height-zero set mode is signalled to the processor, by depressing the "HEIGHT-ZERO SET" push button, which clears I/O bit P8. If the height-zero set mode is active, the 16 bit sensor reading is reduced to 8 bits, and then written to the l.e.d.s on the front panel. (So that the operator can set the switches.) The height reference is read from these switches, and restored to the correctly scaled 16 bit format, before being subtracted from the latest sensor measurement to give the referenced height. This is rescaled and stored in the most significant byte of R3 as HTABS, which is an 8 bit telemetry function.

The software checks if the height lock is active, and if so, fetches the height demand, which is transmitted via the pitch channel. This is re-scaled to the convention of the referenced height signal, and then the height error is determined. The forward gain of the height lock is arranged so that an error of +/- 20.0m will demand the maximum pitch attitude (+/-0.140rad). This is performed, along with the necessary re-scaling gain and offset, to produce a function that is compatible with the original pitch demand convention. This value is checked against the allowable limits, and then stored at R0, to be accessed by the pitch autopilot routine later in the cycle. It is also loaded into the least significant byte of R3 as the telemetry function HTERR. The height routine is completed by writing R3 to location °C3F8°, thus presenting these two telemetry functions.

4.4.3.6 The idle condition.

Once the background software has been completed the processor is put into an idle state until the start of the next System cycle. In this

state, the processor disables its data and address busses, and waits until an interrupt occurs, whereupon control will pass directly to the interrupt handling routine. Once this is completed, the programme will return to the address immediately following that of the idle instruction, which causes a jump back to return to the idle mode. Hence the background software will not be restarted until the eighth cycle section of the servo interrupt routine is performed. This part modifies the return vectors and will prevent a return to the idle loop.

4.4.3.7 The telecommand interrupt routine.

This is the only interrupt that operates in the background, it has priority level 4, and uses the telemetry workspace. It is not used by the foreground software, so there is no possibility of the higher priority servo interrupt routine corrupting the return registers. The flow diagram is given in Fig. 40. R12, the bus read count is cleared, then the telecommand port is read into registers R0 and R1. These are compared to check for any intermittent bus errors. If a fault is high-lighted, the routine will repeat up to five times, (incrementing R12) before aborting. In this condition, TCKNT (R3), which indicates the time since the last telecommand, is not cleared, and no data is transferred. If the read operation is valid the channel address is extracted from the four least significant bits of R1 and then checked to ensure it is less than °B°. This is multiplied by two to accommodate word addressing, and then used to direct the data contained in bits 4 to 11 of R0 to the correct location in the command workspace. On completion, TCKNT is cleared, interrupt 4 is disabled and programme control returns via the stored vectors to the

background software.

4.4.4 The foreground software.

4.4.4.1 The servo interrupt routine.

The programme direction necessary to achieve the foreground software structure is provided by the servo interrupt routine. A flow diagram is given in Fig. 41.

On receiving the level 3 interrupt, which is always enabled, the routine fetches the next output value using the Servo cycle counter, R8, as a pointer. This value is checked against the required range, (0 to 1515), and limited if necessary. The address portion of the output, contained in the four most significant bits, is set-up using R8 and then the complete word is written to the servomotor driving circuit at location °C3F0°, thus initiating the next servomotor pulse. The new cycle flag, in R10, is checked. If this is not zero it indicates that the System cycle has restarted, whereupon the programmable timer, used to generate the Servo period, is reloaded using a multiple bit CRU instruction. This avoids the possibility of a corrupted programmable timer going unchecked by the failure detection circuit, for more than one System cycle. R10 is then cleared, and the clock mode of the programmable timer is re-enabled.

The programme then deals with directing control to the part of the software that will calculate the next servomotor value. R8, the Servo cycle counter, is checked to ensure that it has not been corrupted, and then is used as an index to jump to the relevant branch instruction, which, in turn, routes the software to one of the

control routines. These are independent, having their own workspaces, but they can only return to the background software via the return vectors stored in the servo workspace. Hence when completed, this workspace is restored, and a simple branch is used to return to the servo interrupt routine, which then increments the cycle counter and returns to the next location in the background software. The procedure to clear the cycle counter, and restart the System cycle is described in the next section.

4.4.4.2 The eighth cycle routine.

The flow diagram is shown in Fig. 42. When the Servo cycle count reaches 8, or is corrupted to show a value not in the range 0 to 7, programme control passes to this routine, in order to re-start the System cycle. It does this by altering the stored vectors in the servo workspace to effect a return to the start of the background software on completion. The cycle count is cleared at this stage, and the new cycle flag, in R10, is set (see section 4.4.4.1)

This section also sets up the necessary conditions to allow the servo check interrupt (level 6) to take place. First, the telecommand interrupt is disabled, as these use the same workspace, and so must not be allowed to occur together. The servo check interrupt is enabled, and the interrupt mask in the stored status register in R15, is set to 6. Hence, on returning to the background software, the programme counter points to the System cycle start, and the interrupt mask will allow a level 6 interrupt to be serviced.

4.4.4.3 The servo check interrupt routine.

This is used to check the calibration of the servomotor driving circuit by measuring the duration of a known pulse width against the programmable timer on the c.p.u. card. The flow diagram is shown in Fig. 43. The interrupt routine is initiated by the falling edge of the servomotor driving pulse. Channel 7, which is unused, is set to give the minimum pulse width duration (0.9ms). On receiving this interrupt, the routine temporarily disables the programmable clock (6), and reads the contents of its counter into R6. This is subtracted from the expected value, and a flag, also in R6, is set if the error exceeds the acceptable tolerance of one programmable counter clock period (21.3us). This flag is used by the telemetry routine in the background software to inform the operator of the fault condition. As in the eighth cycle routine, the stored return vectors are modified to point to the start of the background software, but the interrupt mask is now set to 3, thus disabling further servo check interrupts.

4.4.5 The control routines.

The control routines form part of the foreground software. Their function is to calculate the duration of the servomotor driving pulses, a process that includes the control laws for pitch and roll. Chapter 5 deals with selecting these laws, while Chapter 6 is concerned with aspects of microprocessor implementation. Hence the following will be confined to a description of the software logic.

4.4.5.1 The throttle routine.

The flow diagram is given in Fig. 44. This is always under manual control, except during an active parachute deploy condition. The command workspace is loaded, then PFLAG is checked to see if the parachute is to be deployed. If so, the servomotor output register is altered to close the throttle. Otherwise, the SCALE utility routine is used to convert the throttle command in R4 into a form suitable for the output register (R6 in the servo workspace). Programme control returns to the background, via the Servo interrupt routine.

4.4.5.2 The aileron routine.

The aileron is controlled either manually, or via the roll autopilot. The control laws are always carried out, irrespective of the flight mode. This is because they may include digital filter-type compensators, which rely on previous information to formulate the output at any given time. Hence, the discontinuities when changing flight modes are reduced by effectively pre-starting the filter, which is most easily achieved by performing the control laws continuously, and then overwriting the output when in the pilot mode.

The aileron routine, which is shown in Fig. 45, loads the roll workspace, reads the roll attitude sensor and then performs the necessary operations to carry out the control law. This involves scaling the input demand, subtracting the feedback value, and multiplying by the desired forward gain and scaling coefficient. The calculated output is then written to the aileron register (R3) in the servo workspace, having first been converted to the required data format. The command workspace is installed, and then, if the pilot

mode is active, the SCALE utility routine is used to modify this previously calculated value, using the contents of R1 as the direct pilot demand. Programme control then returns as for the throttle routine.

4.4.5.3 The rudder routine.

The rudder is operated manually, though subsequent systems may include yaw control. Hence a rate gyroscope has been included in the instrumentation, but the software only caters for the pilot mode. The yaw and heading workspaces are unused at present, as the rudder routine operates in the command workspace. SCALE is used to convert the pilot demand in R3 to the servomotor output format. The result is placed in R4 of the servo workspace. The flow diagram is shown in Fig. 46.

4.4.5.4 The elevator routine.

This uses the pitch workspace, and is shown in Fig. 47. The routine first checks for the height loop, which, if active, will have prepared a pitch attitude demand in the height data preparation routine (section 4.4.3.5). Otherwise the demand is fetched from the command workspace and converted to the signed, fixed point standard used in the control law. The attitude sensor is read using SENS, then the digital filter routine is performed, using the sensor reading as its input, which places the compensator in the feedback path. The result, referred to as the modified feedback, is subtracted from the re-scaled demand and multiplied by the forward gain and scaling coefficient. The structure of the software implementation of the

filter is discussed in section 6.5. The output value is stored in R5 of the servo workspace, then the routine checks for the pilot mode. As before, if this is active, the servomotor demand is re-calculated, using SCALE with R2 as the input.

Part II DESIGN OF THE FIXED GAIN CONTROL SYSTEM.

Chapter 5 The design of the fixed gain autopilot control laws.

5.1 Summary of previous work.

This project was started at a time when the Stabileye programme was undergoing a period of considerable development and change. This process culminated in the design of a new airframe (referred to as the Mk.3), and the definition of a new analogue flight control system. Prior to this, the control electronics had been haphazard, both in their design and execution. This was mainly due to attempts to use low-cost sensors (13), which had proved to be too unreliable for practical purposes. Thus the new analogue controller used conventional aircraft-standard sensors, notably a dual-axis attitude gyroscope for pitch and roll. At the same time, the autopilot configuration was defined as:

Active roll attitude control via the ailerons.
Active pitch attitude control via the elevators.
Active height control via pitch attitude.

Experience with the Mk.1 airframe had suggested that active yaw control would not be necessary. The design of these later control laws is given elsewhere (14).

Originally the DFCS was installed in an earlier Mk.1 airframe, but difficulties surrounding the deployment of this increasingly obsolete aircraft caused successive postponements of its first flight throughout 1981. So in mid-1982 it was decided to reconfigure the system for the Mk.3 airframe. By this time the analogue controller described above was well-established, and it

was apparent that the fixed gain version of the DFCS should attempt to mimic this.

5.2 Representation of the aircraft dynamics.

A full treatment of this topic, as presented in texts on aerodynamics (15), would not be relevant here. For the purposes of this study, the dynamics of the aircraft have been reduced to the de-coupled, fixed velocity, short period transfer functions, relating roll rate to aileron deflection, pitch rate to elevator deflection and height to pitch attitude. These have been derived from estimates of the stability derivatives based on wind tunnel measurements, (16). The transfer functions were evaluated at 22, 40 and 50 m/s, to give the results shown in Table 4. These are similar, but not identical, to those used in the analogue autopilot design (14) as subsequent work has led to revised estimates of certain derivatives. The particular airspeeds chosen represent the slowest safe operating speed, a typical cruising speed, and the maximum speed of the aircraft. These are somewhat arbitrary, and depend upon the trim condition, weight and engine installation employed for a particular flight.

The transfer functions all have negative steady state gain. This is due to the sign conventions adopted for the control surfaces, and the resulting movements of the aircraft. Fig.48 shows the positive senses of the control surface deflections, the Body axes and the Body rates. The Body axes form a right-handed set, with their origin at the centre of mass, and OX aligned with the direction of motion. The Body rates (p, q and r) describe the

components of angular velocity with respect to inertial space, taken along the instantaneous directions of the OX, OY and OZ axes. These are the quantities that would be measured by suitably aligned rate gyroscopes mounted in the aircraft.

The analysis assumes that the aircraft is initially in an un-accelerated, steady state condition, with wings level and no sideslip. After a disturbance the Body axes will assume different positions. The relation of these to the steady state can be derived by three rotations:

Through , an angle of pitch; θ

through , an angle of bank; ϕ

and through , an angle of yaw. ψ

The aircraft has angular velocities $\dot{\theta}$, $\dot{\phi}$ and $\dot{\psi}$. Strictly speaking these are not equal to p, q and r as they are not the components about OX, OY and OZ. However, for small distrubances as assumed in the following analysis, this discrepancy can be ignored. Thus, the relationships of control surface deflections to aircraft attitudes can be derived from the expressions in Table 4, by integration, with zero initial conditions.

While taking these transfer functions as representative, it is prudent to consider the assumptions made in obtaining them. These are:

- The mass of the aircraft is constant during any particular dynamic analysis.
- The aircraft is a rigid structure.

- The earth is the inertial reference, and the atmosphere is fixed to this.
- OX and OY lie in a plane of symmetry, so that the products of inertia, I_{xy} and I_{yz}, are zero.
- Motion is restricted to small deviations from the initial steady-state.

This last condition has the effect of linearising and de-coupling the equations of motion, so that the design process can deal with the roll and pitch loops separately.

5.3 The design objectives.

The objective was to obtain a system which gave an acceptable combination of handling qualities and relative stability. The former being a compromise of the speed of response and the degree of oscillation involved. The best available guideline was the intended step responses of the continuous analogue system ((14), Figs. 5 and 8), which are reproduced in Figs. 49 and 50. It will be shown that these were obtained using an unrealistic model of the control surface and servomotor. Therefore they should not be taken as necessarily achievable, but they do serve to illustrate the required level of damping.

In the light of this, the design criteria were:

- to use a gain less than one third of that which causes instability.
- to reproduce the percentage overshoot of Figs. 49 and 50,

at the cruise speed of 40 m/s.

- to ensure that the step responses at 22 and 50 m/s are acceptable, albeit less so than the cruise speed case.

5.4 Z-plane analysis of the roll and pitch loops.

5.4.1 Deriving a consistent mathematical representation.

A sampled data control system is one in which some, but not all of the signals appear as a sequence of numbers. This presents difficulties to Laplace transform methods, as it does not result in rational functions of s. A mathematical tool developed for the analysis of such systems, is the z-transform (17) (18). The role of the z-transform in discrete systems is similar to that of the Laplace transform in continuous systems. Briefly, F(z), the z-transform of f(t) is given by:

$$F(z) = \sum_{k=0}^{\infty} f(kT)\, z^{-k} \qquad 5.1$$

where $z = e^{sT}$, and T = the sampling period.

This is an infinite series and requires additional effort to obtain the closed form. For this reason, it is usually more convenient to obtain F(z), for a given f(t) or F(s), by consulting tables of transforms (17) (18) (19). The following example discusses the z-transform treatment of the roll or pitch loop.

Fig.51 shows an autopilot loop as configured in the DFCS. Note here the distinction between a discrete and continuous time variable, and likewise, an analogue or digital signal. This must

be reduced to its mathematically significant operations, to allow the use of sampled data theory. These are shown in Fig.52.

It has been assumed that the sensors have constant gain, and bandwidths very much greater than those of the control loops. Thus they have no significant effect, beyond the need for suitable re-scaling.

The digital computations use at least 12 bits, so the quantization errors resulting from digital representations of analogue signals are insignificant, and the action of the ADC and the DAC need not consider this particular aspect. Therefore the ADC is reduced to a sampler, with a sampling aperture that is very small compared with the time constants involved with the system. This can be represented by a switch, which closes and opens instantaneously, giving the following model:

$$e(t) \quad \text{——/——} \quad f(kT) \qquad 5.2$$

$$\text{where } f(kT) = e(t)\Big|_{t=kT}$$

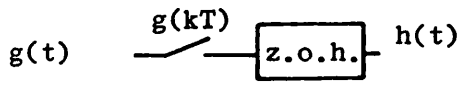
The digital computer operates on the number sequence, f(kT), to provide another number sequence, g(kT). It is the relationship of these two that the design process is attempting to determine. The control law could be a simple gain or a series of difference equations forming a digital filter (compensator). Either may be expressed as a discrete transfer function D(z), so the computer is described by:

$$G(z) = D(z)\,F(z) \qquad 5.3$$

where G(z) is the z-transform of the output sequence,

and F(z) is the z-transform of the input sequence.

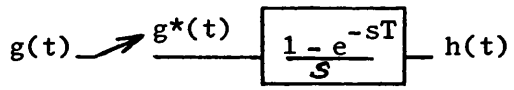
The DAC and zero-order hold require special treatment, namely the use of impulse sampling (17) (18). The zero-order hold is described by the following:



5.4

where $h(kT + \tau) = g(kT)$, for $0 \leq \tau \leq T$.

This can be shown to be equivalent to:



5.5

where $g^*(t) \triangleq \sum_{k=-\infty}^{\infty} g(t)\ \delta\ (t - kT)$

This is a sequence of weighted Dirac delta (impulse) functions, separated by T-second intervals, and is not physically realisable. Its importance is that it replaces the hybrid zero-order hold with the continuous transfer function T(s), where:

$$T(s) = \frac{1 - e^{-sT}}{s} \qquad 5.6$$

Thus the roll loop can now be represented as shown in Fig.53. The transfer functions describing this system are:

$$\frac{F(z)}{E(z)} = D(z)\ \mathcal{Z}\left[\left(\frac{1 - e^{-sT}}{s}\right)\left(\frac{C_R(s)}{s}\right)\left(C_{sv}(s)\right)\right] \qquad 5.7$$

and $E(z) = R(z) - F(z)$.

Applying the shifting theorem (17) (18), these can be reduced to:

$$\frac{F(z)}{E(z)} = D(z)\,(1 - z^{-1})\; \mathcal{Z}\left[\frac{(C_R(s))\;(C_{SV}(s))}{s^2}\right] \qquad 5.8$$

and $E(z) = R(z) - F(z)$.

The z-transform of the function of s in equation 5.8 can be obtained by reference to tables, to give:

$$\frac{F(z)}{E(z)} = D(z)\,(1 - z^{-1})\; G_{RSV}(z) \qquad 5.9$$

and $E(z) = R(z) - F(z)$.

This is now composed entirely of functions of the complex variable z, and can be treated by one of several design techniques. The z-plane root locus (17) was chosen, as this can be carried out directly from equation 5.9, whereas frequency domain methods require a further transformation.

The root locus was used to suggest the form of the controller. That is, whether a simple gain was adequate, or if a compensator was necessary. If so, the type of compensation could be deduced. Following this, simulations of the roll and pitch loops were prepared to confirm the results (see Section 5.6).

5.4.2 Obtaining the z-plane root loci.

5.4.2.1 Z-transformation of the aircraft transfer functions.

Again, consider the roll loop design. This requires the evaluation of:

$(1 - z^{-1})\; G_{RSV}(z)$ (part of 5.9).

Now:
$$G_{RSV}(z) = \mathcal{Z}\left[\frac{1}{s}\,(C_{SV}(s))\,\frac{(C_R(s)}{s}\right] \qquad 5.10$$

$C_R(s)$ is the aerodynamic transfer function between the control surface deflection and the Body rate, given in Table 4. $C_{SV}(s)$ is the transfer function of the servomotor and control surface sub-system. This was taken as: (see Section 5.6.2)

$$C_{SV}(s) = \frac{20.0}{s + 20.0} \qquad 5.11$$

The expression in equation 5.10 has a fourth order denominator, in the case of the roll loop, and fifth order for pitch. It is therefore unlikely that they could be found in tables of z-transforms, in this form. Now, in general, it is not possible to take z-transforms of each factor of s in the expression in 5.10, as:

$$\mathcal{Z}\left[C_1(s)\,C_2(s)\right] \neq G_1(z)\,G_2(z) \qquad 5.12$$

where $G_1(z) = \mathcal{Z}\left[C_1(s)\right]$ etc.

So, it is necessary to subdivide the expression using partial fraction expansion. This gives a sum of first and second order functions of s, that can be transformed more readily using tables. The result (a sum of similar functions of z), is not as useful as would be a corresponding product of factors. This latter form would allow cascade implementation of the discrete transfer function at the simulation stage. Hence the sum has to be reduced to a single quotient, and then factorised.

This procedure had to be performed a number of times. It is cumbersome, and includes calculations that exceed the accuracy capabilities of a hand calculator. A computer programme was developed, which handled the algebraic manipulations described above, using a numerical analysis package (E1). The programme is called ZTRANS.FORTRAN, and is listed in Appendix E, along with details of its use. The results produced are given in Table 5.

5.4.2.2 Constructing the loci.

Now, the characteristic equation of the system in Fig.53 is given by:

$$1.0 + D(z)\ (1 - z^{-1})\ G_{RSV}(z) = 0 \qquad 5.13$$

The expression $(1 - z^{-1})\ G_{RSV}(z)$ has been evaluated by ZTRANS.FORTRAN, and D(z) is the discrete transfer function of the digital computer control law. Until the nature of the control action is determined, D(z) is taken to be a simple gain, K. Hence the root locus is obtained by repeated solution of:

$$1.0 + K\ (1 - z^{-1})\ G_{RSV}(z) = 0 \qquad 5.14$$

as K varies.

An approximate locus could be constructed using the same rules as those for a continuous system, but this would not indicate the precise value of K for a particular set of pole positions. For this reason, another computer programme making use of the numerical analysis package (E1) was developed, to provide the roots of equation 5.14. This programme, GENLOC.FORTRAN, is presented in

Appendix E, along with details of its use. The results were produced in tabular form, allowing the root loci to be drawn.

5.4.2.3 Results for the roll controller.

The root locus for the roll loop, using the 40 m/s airframe transfer function, and the first order model for the control surface subsystem, is given in Fig.54. Root positions for certain significant values of gain, as produced by GENLOC.FORTRAN, are given in the Fig. The following points can be observed:

- The response will be dominated by the roots of the branch of the locus associated with the open loop poles at $z = 1.0$ and $z = 0.6125$.
- These roots are complex for $K > 0.35$.
- The stability boundary is crossed in the region $3.0 < K < 4.0$, (so the design objective requires that $K < 1.0$).
- Assuming that a damping ratio of approximately 0.8 is required, indicates that the forward gain, K, should be 0.45. Compensation will not be necessary.

5.4.2.4 Results for the pitch controller.

The root locus for the pitch controller with a simple forward gain, K, and a first order control surface model, is given in Fig.55. From this it can be seen that:

- The response will be dominated by the contributions of a slow, real pole and a complex pole pair, for which the

damping ratio is always less than 0.35

-The stability boundary is crossed for K> 2.0

-If the gain is increased to reduce the time constant of the real pole, the oscillations due to the complex pair will be unacceptable. Clearly some form of compensation is required.

5.5 Choice of the pitch compensator.

It was decided to use the equivalent of the compensator used in the continuous analogue scheme (14). This relies on pole-zero cancellation to reduce the contributions of the undesired open loop roots. Inexact cancellation is inevitable, especially at 22 and 50 m/s. This may lead to unacceptable step responses, or even instability, but these will be examined at the digital simulation stage.

The transfer function of the compensator is given below, as a function of s and then as the equivalent function of z:

$$C_C(s) = \frac{2.02(s^2 + 8.0s + 80.0)}{(s + 4.0)(s + 39.6)}$$

and: 5.15

$$G_C(z) = \frac{1.3272(1.0 - 1.7736z^{-1} + 0.8187z^{-2})}{(1.0 - 0.9048z^{-1})(1.0 - 0.3714z^{-1})}$$

An equivalent function could have been achieved by a number of discretization methods, such as z-transformation with, or without a hold, bilinear transformation or root matching (20). The

different techniques would produce varying results, as each method preserves some, but not all, of the analogue original. A comparison of these is given in the references (23); the latter method was chosen. Root matching maps the poles and zeroes of $C_c(s)$ on the s-plane to equivalent positions on the z-plane. As pole-zero cancellation is the object, this is the most suitable approach.

Discretisation is as follows:

a) Replace factors of the form $(s + a)$ with $(1.0 - e^{-aT}z^{-1})$.

b) Replace factors of the form $(s+a \pm jb)$ with

$$(1.0 - 2.0e^{-aT}\cos(bT)z^{-1} + e^{-2aT}z^{-2}).$$

c) Re-scale the steady state gain.

The poles and zeroes of the pitch compensator are shown on the root locus diagram in Fig. 55. Note that a similar pole-zero cancellation compensator could have been derived directly by inspection of this z-plane root locus. The intention to reproduce the analogue scheme control law exactly, deterred this.

5.6 Discrete simulation of the roll and pitch loops.

5.6.1 The simulation method.

The control surface and servomotor subsystem will be referred to as the actuator. The model of this actuator, used in the estimation of the root loci, was a linear, first order transfer function (see equation 5.11). This will be shown to be inadequate for other than small deflections. The purpose of the discrete simulation was to confirm the control laws suggested by the root loci, at 22, 40 and 50 m/s, using a more representative actuator model. So, the

actuator dynamics, previously embedded in the continuous part of the system in Fig.53, must now be separated from the aerodynamic transfer function.

The simulation method was to precede each separate continuous section of the loop with a sample and hold block. In this way, they can be represented as discrete transfer functions in a digital computer programme (17). For example, the functional block diagram of the simulation of the roll loop is given in Fig.56. (The treatment of the pitch loop was similar.) Z-transformation was achieved, as before, using ZTRANS.FORTRAN (see Section 5.4.2.1). The actuator model will be considered first.

5.6.2 The rate-limited actuator model.

5.6.2.1 Earlier actuator models for Stabileye.

Previous autopilot design studies for Stabileye have used a second order linear model for the actuator, of the form:

$$\frac{\text{Output position}}{\text{Demand}} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \qquad 5.16$$

This type of model arises by measuring the frequency response of an unloaded servomotor. Results of this nature were obtained, which suggested that $\omega_n = 19.0$ rad/s and $\zeta = 0.65$, (Ref.(14) used $\omega_n = 20.0$ rad/s, $\zeta = 0.5$).

When considering the unit step response of this type of model, it becomes clear that it is not representative. In particular, the percentage overshoot suggested by 5.16 is given by:

$$100.0 \times \exp\left(\frac{-\Pi\zeta}{\sqrt{1-\zeta^2}}\right) \qquad 5.17$$

(For $\zeta = 0.5$, overshoot = 16.3%)

However, practical experimentation reveals that the servomotor does not exhibit any noticeable overshoot. It was therefore decided to carry out a more thorough investigation of the actuator, with special attention to loading and rate limiting effects.

5.6.2.2 Servomotor loading.

The most significant loading effects will be due to aerodynamic forces, as the control surfaces are lightweight structures, with low hinge-friction. This type of loading has been measured for Stabileye, and was found to be dependent on incidence and airspeed (21). The maximum values were recorded with the control surfaces at their full deflection. They were:

0.105 Nm for the elevator (24 m/s),

and 0.18 Nm for the aileron (30 m/s).

These airspeeds are lower than would be expected under normal flight conditions, but there is no other data available.

The mechanical advantage of the linkages must also be considered. As driven by the DFCS, the servomotor outputs will move through 1.798 rad. The desired elevator travel is 0.524 rad, the aileron 0.349 rad, so the linkages afford mechanical advantages of:

3.43 : 1 for the elevators,

and 5.15 : 1 for the ailerons.

The maximum load, under which the Skyleader SRC4BB servomotor can move, is 0.34 Nm (22). Thus the 'worst case' conditions quoted above provide only 9% and 10.2% of the stalling load. Hence loading effects need not be included in the actuator model, as typical working loads are much less severe than these.

5.6.2.3 Rate-limiting and deadband.

Rate-limiting, however, is significant. This is confirmed by Fig.57, which shows the response of a servomotor to step changes of input demand. (More specifically, to changes in the duration of the periodic driving pulses.) Increments of more than 0.125 ms give responses that are clearly dominated by fixed rate regions. This particular servomotor shows a limit of 4.014 rad/s, while the specified output rate limit is 3.490 rad/s (22). To be meaningful this must be related to the geometry of the control surfaces. Taking into account the reduction due to the linkages, and using the specified rate of 3.490 rad/s, the limits are:

0.678 rad/s for the elevator,

and 1.018 rad/s for the aileron.

The 0.0625 ms step shown in Fig.57 is more reminiscent of a linear system. This is expanded in Fig.58, which shows the responses to smaller changes of input pulse duration. These can be adequately described by a first order lag, with a time constant of 0.05 s. (Hence the linearised, small deviation model used in the root locus analyses.)

The smallest step in Fig. 58 is uncertain, and does not reach the demanded position. Subjecting the servomotor to changes in demand that were smaller than this did not result in any output movement at all. This is due to stiction in the mechanism and backlash in the drive gears, giving rise to a deadband region.

5.6.2.4 Description of the non-linear actuator model.

The form of the state equations necessary to model the actuator and aircraft dynamics will be as described in section 5.6.3. Sections 5.6.2.2 and 5.6.2.3 have shown that servo-actuator loading effects are not significant, but position and rate limiting and deadband are. It is therefore necessary to modify the state equation corresponding to the actuator, (as shown in Fig. 60 for the roll loop). Deadband is simulated by modifying the servo-actuator demand function. If the absolute value of the difference between the latest and the previous servo-actuator demands is less than the specified deadband level, the latest demand is replaced by the previous demand.

The model then checks for rate limiting. If the difference between the latest servo-actuator position (designated XS) and the previous position (XSO), is such that it exceeded the rate limit in order to achieve this change, then the value of XS is altered to:

$$XS = XSO +/- (\text{Rate limit})(\text{Iteration interval}) \qquad 5.18$$

The next process is to ensure that the servo-actuator position, as calculated by the state equation corresponding to the actuator, or as

modified by equation 5.18, does not exceed the position limit. If it does, it is replaced by that value.

The model can mimic the extent of the non-linear effects shown in Figs. 57 and 58, or be set to represent the ideal actuator. The measured values of position and rate limits were:

Aileron, position = +/-0.175 rad, rate = +/-0.678 rad/s
Elevator, position = +/-0.262 rad, rate = +/-1.018 rad/s 5.19

Deadband was set so as not to respond to less than 2 bits change at the input demand, i.e. the deadband value was:

(Total actuator travel)/128 rad 5.20

5.6.3 Computer implementation of discrete transfer functions.

The general roll and pitch transfer functions for attitude, without the actuator dynamics are: (as provided by ZTRANS.FORTRAN)

$$G_R(z) = \frac{K(z + a)}{(z - 1.0)(z + b)} \quad \text{and} \quad G_P(z) = \frac{K(z + a)(z + b)}{(z - 1.0)(z^2 + dz + e)} \qquad 5.21$$

These lend themselves to cascade implementation of simple first and second order components. So that:

$$G_R(z) = (K)\left(\frac{1.0}{z - 1.0}\right)\left(\frac{z + a}{z + b}\right) \qquad 5.22$$

$$G_P(z) = (K)\left(\frac{1.0}{z - 1.0}\right)\left(\frac{(z + a)(z + b)}{(z^2 + dz + e)}\right) \qquad 5.22 \text{ (cont.)}$$

For physical interpretation these must be expressed as functions of z^{-1} which is equivalent to a delay of one sample period. So:

$$G_R(z^{-1}) = (K)\left(\frac{z^{-1}}{1.0 - z^{-1}}\right)\left(\frac{1.0 - az^{-1}}{1.0 - bz^{-1}}\right) \qquad 5.23$$

$$G_P(z^{-1}) = (K)\left(\frac{z^{-1}}{1.0 - z^{-1}}\right)\left(\frac{1.0 + (a+b)z^{-1} + abz^{-2}}{1.0 + dz^{-1} + ez^{-2}}\right)$$

The computer programme realisations can be inferred directly from these equations(23), or by first drawing the state diagrams(17). The derivation of the difference equations for the roll loop will be described here as an example. The block diagram is given in Fig. 56, (for later work, in part III, a fixed disturbance input was added at the output of the actuator). The state diagram for the forward path of the roll loop simulation is given in Fig. 59, showing the servo-actuator and disturbance inputs and the error, rate and acceleration outputs. The labels in this diagram correspond to the names used in the difference equations in the simulation programme. The values of these coefficients can be derived from Tables 5, or 5a, in the rows marked $G_R(z)$ and $G_{SV}(z)$. (There is a requirement for two sets of coefficients because the simulation was later repeated with a much higher sampling rate than 40.0Hz).

Fig. 60 shows the difference equations taken from the programme listing. The states XS,XE1,XE2 etc. are at the outputs of the paths marked z^{-1} in the state diagram. The one sample period delay is effected by calculating the next value of the state XS1, XE11, XE21 etc. and

storing that value until the next iteration, wherein it is passed to the current value of the state.

5.6.4 Notes on the simulation programmes.

Separate simulations for the pitch and roll loops were produced. These were titled ZROLL.FORTRAN and ZWORKS.FORTRAN, and are run interactively on a PDP11/34 minicomputer. A sample listing of ZROLL.FORTRAN is given in Appendix F. Many of the options included are only necessary for the adaptive autopilot to be investigated in part 111. This accounts for the need to simulate the rate and acceleration functions for roll, as shown in Fig. 56. (These are fed back via switched gains in the adaptive scheme.) Both programmes have definition, simulation and output stages. The first allows the operator to define a switching surface (see part 111), set-up the servo-actuator parameters, select "true" or "estimated" rate (the latter includes measurement noise and is derived from a digital filter), and set-up the gains and compensator coefficients. The simulation stage generates the step response over a 4.0s period, with either zero initial conditions and a non-zero demand, or vice versa. The difference equations for actuator and aircraft dynamics originally used a 40.0Hz sampling rate. This was considered adequate after comparing results with those from a 4th order Runge-Kutta algorithm with a 0.01s iteration interval. The difference was not significant, and by using this sampling rate the servomotor update rate would be simulated directly. However, in later work where high frequency switched gains were used, it was found that the transport

delay of the z-transfer function models interfered with the performance. Hence the roll simulation only was repeated with a 320.0Hz sampling rate for the dynamics, but with the control signal still updated at 40.0Hz. This is very much faster than the dynamics of the system and resulted in z-transfer function coefficients that were close to 1.0. The accuracy of the minicomputer arithmetic was considered adequate to accommodate this problem. The actuator and roll z-transfer functions at this higher rate are given in Table 5a.

The output stage allows the operator to display the attitude, the servo-actuator displacement or the switching function. It also provides the option to change any of the of the previously defined parameters, before repeating the simulation stage.

5.6.5 The roll attitude simulation results.

The gain of 0.45 suggested by the root locus was reduced to 0.4, as this gave better results. These are shown in Figs. 61 to 66, which depict the attitude and actuator movement for a step demand of 0.5 rad, at 40, 22 and 50 m/s. The desired shape of response at 40 m/s has been achieved. The appearance of overshoot in Fig. 61 is caused by the actuator deadband condition. This gives rise to a small, non-zero steady state value, which results in a certain amount of attitude drift. The other non-linear servomotor aspects are clearly seen, and have a significant effect on the transient behaviour of the attitude. The step responses at 22 and 50 m/s are acceptable, so the only remaining design objective concerns the relative stability. This was checked by examining the performance with a forward gain

three times as large as the chosen value. The step response and the aileron results for the worst case of 50 m/s are shown in Figs. 67 and 68. These are still convergent.

A simulation using the linearised second order actuator model of equation 5.16 was also developed, to illustrate its inadequate behaviour. The results for the same step demand, with a gain of 0.4, are shown in Figs. 69 and 70. The response is 'faster' and does not exhibit any steady state error. This explains the apparently superior performance of the analogue controller in Fig.49. However, when the actuator displacement, as shown in Fig.70, is examined it is seen to be unrealistic, with a maximum deviation of 0.3 rad and a maximum rate of 2.5 rad/s.

5.6.6 The pitch attitude simulation results.

The second order compensator suggested in Section 5.4.2.4 was tried in both the forward and feedback paths. It was found that while both were subject to similar degradation in performance at 22 and 50 m/s, the feedback compensation gave generally shorter transient periods. The difference in the two configurations is confined to the closed loop zeros. Using forward path compensation these are the zeros of the plant and the compensator. With feedback compensation, they are given by the plant zeros, but the compensator poles. Being faster in its responses, the latter configuration was chosen, along with a forward gain of 0.86.

The attitude response, and actuator displacements at 40, 22 and 50 m/s, for a 0.14 rad step demand, are given in Figs. 71 to 76.

The transient shape of the 40 m/s case is similar to the objective shown in Fig.50. The performance at 22 and 50 m/s shows the effects of inexact cancellation. However, when the amount by which the plant parameters change is considered, the responses at the extremes of the range are not unacceptable, provided the relative stability is not impaired. This was examined at all three airspeed values. The worst case, at 50 m/s, is shown in Fig.77. Though this suffers almost 100% overshoot, it settles to a steady state in 1.5 s, and is stable.

Actuator excursions are smaller than for the roll simulation, so position and rate limiting are not encountered. This does mean that the deadband effect is worsened, as it constitutes a more significant proportion of the working range of the actuator.

5.7 Treatment of the height loop design.

The continuous autopilots had shown that the height loop was acceptable without compensation. As the sampling period is very much shorter than the time constants involved, it was decided that the conversion to a sampled-data scheme would have no effect on its performance.

Had the height loop required discrete compensation, this approach could not have been used. This is because compensator time constants that are much longer than the sampling period, result in z-plane poles and zeroes that are close to z = 1.0. When this happens, the errors arising from fixed point quantisation noise lead to instability.

The gain chosen was:

$$K_H = 0.1939$$

This corresponds to a linear range of $\pm$ 20.0m with respect to the desired height. Beyond this the pitch attitude loop saturates, so that the aircraft performs a fixed rate of ascent or descent.

Chapter 6 Microprocessor implementation of the control laws.

6.1 Choosing the finite wordlength notation.

The design process in Chapter 5 has defined the gains and compensation that will give the desired performance. These have been implemented in the digital simulation and must now be transferred to the simpler arithmetic of the microprocessor system. This entails choosing the finite wordlength, and the manner by which values should be represented. The former is intimately related to the hardware, so this aspect was considered during the selection of the c.p.u. and the A/D converter, which are 16, and 12 bit devices respectively.

The next step is to select the number system. The choice is between fixed and floating point notation (23, 25). The latter requires that each value be expressed by a mantissa and an exponent, and must complicate the software. As the amount of c.p.u. time available for the control algorithms was not known, it was decided to use fixed point notation which would be more straightforward.

6.2 Hazards of fixed point arithmetic.

The principal drawback of this system is its limited dynamic range. This can cause difficulties due to quantisation noise, if the wordlength is not adequate. Alternatively, if the result of a calculation exceeds the permitted range of values, a condition known as 'overflow' will occur.

It is clear that the likelihood of these problems will depend on how the arithmetic is scaled. Providing the 12 and 16 bits of the

hardware are used carefully, it was felt that quantisation errors would not be significant. Their effect was examined empirically by the use of a hybrid simulation, to be described in Chapter 7. Overflow is more serious as it gives rise to completely incorrect results, and so should be dealt with actively. This was done by checking after each addition, or multiplication, and replacing an erroneous result with the maximum positive or negative number allowed. Using this method, referred to as saturation (25), overflow is still undesirable, but less likely to be catastrophic.

6.3 Treatment of multiplication.

In fixed point arithmetic, the product of two 16 bit numbers is a 32 bit wordlength. This must be expressed as a 16 bit result by truncation, or better still, rounding off. It is necessary to ensure that the abbreviated result is scaled to the same standard as the multiplicand. To achieve this, the multiplier is prepared in floating point notation, i.e. as:

$$K \times 2^{n} \qquad 6.1$$

where K is a positive 16 bit binary fraction, and n is a positive exponent, used to pre-shift the multiplicand. The multiplication process itself would not normally cause overflow; it is this shifting procedure that might. The routine accommodates negative numbers by performing two's complement negation of the result.

6.4 Scaling.

While the possibility of overflow could have been avoided entirely by cautious scaling, it was felt that the attendant loss of accuracy

in the calculations would be a greater problem. As the effect of overflow is actively catered for in the software, it might be allowed to occur during extreme manoeuvres in flight. The same reasoning was applied to the adjustment of the sensor outputs to the A/D converter input, and so the hardware scaling was duplicated in the software.

The standard for each loop is obtained by equating the full scale sensor output, to the maximum value returned by the SENS sub-routine. Now, SENS multiplies the 12 bit ADC output by sixteen, so these maximum values are:

$$\pm\ 2047 \times 16 = \pm\ 32752 \qquad 6.2$$

Thus the scaling standards for the control law calculations are as follows:

$$\begin{aligned} &\text{Roll,} && \pm\ 1.396 \text{ rad} && = && \pm\ 32752 \\ &\text{Pitch,} && \pm\ 0.291 \text{ rad} && = && \pm\ 32752 \qquad 6.3 \\ &\text{Height,} && \pm\ 1000 \text{ m} && = && \pm\ 32752 \end{aligned}$$

The demands are all expressed as 8 bit numbers, as they occupy a byte in the telecommand frame. So, initially they are scaled to:

$$\begin{aligned} &\text{Roll demand,} && 0 \text{ to } 255 && = && -1.047 \text{ rad to } +1.047 \text{ rad} \\ &\text{Pitch demand,} && 0 \text{ to } 255 && = && -0.140 \text{ rad to } +0.140 \text{ rad} \\ &\text{Height demand,} && 0 \text{ to } 255 && = && 0 \text{ to } 1000 \text{ m} \qquad 6.4 \end{aligned}$$

These must be adjusted to the sensor standards of expression 6.3, before they can be used in the control laws.

The outputs of the roll and pitch calculations are control surface deflections. Because of the dissimilar geometry of the aileron and elevator linkages, these require different scaling factors, to provide the numbers expected by the servomotor driving circuit (see Section 3.2.3.3). The output scaling standards are:

Aileron servomotor, 0 to 1515 = -0.175 rad to +0.175 rad

Elevator servomotor, 0 to 1515 = -0.262 rad to +0.262 rad

6.5

The output for the height loop is converted to the pitch demand scaling given above.

With the information in expressions 6.3, 6.4 and 6.5, it is possible to assemble the multiplication factors used in the fixed point routines. These are the coefficients referred to in the flow charts in Figs. 45, 46, 47. An example will be given to illustrate how they are derived. Consider the pitch loop forward gain, which is labelled PFGSC in the software. The specified gain is 0.86, but this must also accommodate the re-scaling necessary for the output standard. The result of the pitch summing junction will be scaled to:

$$\pm\ 0.291 \text{ rad} = \pm\ 32752 \qquad 6.6$$

The output standard is:

$$\pm\ 0.262 \text{ rad} = \pm\ 757.5 \qquad 6.7$$

So the overall multiplication coefficient will be:

$$0.86 \times \frac{757.5}{0.262} \times \frac{0.291}{32752}$$

$$= 0.022105 \times 2^{0} \qquad 6.8$$

$$= °05A8° \text{ , expressed as a binary fraction.}$$

The derivations of all the control algorithm coefficients are given in Table 6.

6.5 Structure of the pitch compensator.

The only remaining part of the control loops requiring explanation is the structure of the pitch compensator. This was given in equation 5.15 as:

$$G_c(Z) = 1.3272 \quad \frac{(1.0 - 1.7736Z^{-1} + 0.8187Z^{-2})}{(1.0 - 0.9047Z^{-1})(1.0 - 0.3715Z^{-1})} \qquad 6.9$$

This can be re-arranged to the form:

$$G_c(Z) = GN \quad \frac{(1.0 + A1Z^{-1} + A2Z^{-2})}{(1.0 + B1Z^{-1} + B2Z^{-2})} \qquad 6.10$$

The software implementation of this follows the method used to produce discrete transfer functions in the digital simulation, (Section 5.6.3). The state diagram is shown in Fig. 78, and the calculations for the (i + 1)th iteration are given below.

$$X_1(i+1) = GN\ R(i) - B1\ X_1(i) - B2\ X_2(i)$$

$$X_2(i+1) = X_1(i) \qquad 6.11$$

The output equation is:

$$C(i) = GN\ R(i) + (A1 - B1)X_1(i) + (A2 - B2)X_2(i)$$

where R(i) is the pitch attitude input, and C(i) is the modified feedback.

A listing of the assembler software is given in Appendix C, and the values of the multiplier coefficients SSGN, PCGB1, PCGB2, PCGAB1 PCGAB2 are shown in Table 6. These are stored contiguously in memory so that it is only necessary to load the multiplier pointer in R12 at the start of the routine. Thereafter, the pointer is incremented automatically, each time that MULT is called. The last stage of the compensator routine moves the newly calculated states $X_1(i+1)$ and $X_2(i+1)$, to the X_1 and X_2 register stores in readiness for the next iteration.

Chapter 7 The fixed-gain autopilot results.

7.1 Hybrid simulation.

7.1.1 The method.

Hybrid simulation is a technique where the DFCS is operated in closed loop with an analogue computer model of the aircraft dynamics. It offers the opportunity to study the performance of the hardware system at first hand, with the hope of eliminating implementation errors before flight trials. These errors fall into three categories:
1) Noise problems, including quantization. The analysis in Chapter 5 examined the effect of using a discrete-time control system, but ignored the quantization of signal levels. By using the actual input/output systems of the DFCS, representative levels of measurement noise are introduced.
2) Faulty logic in the micro-code. Corrected by editting and re-assembling the software. This exploits the inherent versatility of a processor-based system.
3) Hardware errors of the integrated system. These are typically interconnection faults which were not discovered during individual bench testing of the circuitry.

These tests were carried out during the design of the DFCS for the Mk 1 Stabileye. Having verified its performance, and eliminated all of the above errors, it was not felt necessary to repeat this stage when the Mk 3 Stabileye control laws were introduced. Therefore the material given here corresponds to the earlier autopilot configuration.

The most complete hybrid simulation would be one that excercised

the entire airborne DFCS. For practical reasons, certain parts of the hardware could only be simulated, as shown in the block diagram in Fig. 79. A brief description follows, with a more comprehensive treatment of the analogue computer modelling of the aircraft transfer function given in Appendix G.

The sensors were omitted as there was no means of moving the gyroscope, or providing a barometric input. Therefore the analogue computer models were scaled to produce outputs which duplicated those of the instruments. It was also necessary to replace the telecommand receiver with a test set which simulated the receiver's output. This unit allowed the operator to generate precise step demands for the pitch and roll loops. The deflection of the actuator was measured by driving a potentiometer, the output of which was used to stimulate the analogue computer model. (This arrangement was used for the actuator rate-limiting investigation in section 5.6.2.3)

7.1.2 Hybrid simulation results.

Fig. 80 shows the hybrid simulation results for the Mk 1 Stabileye roll controller. This aircraft operated at much slower airspeeds than the Mk 3 version, so the aircraft transfer functions for 22 and 30m/s were used. Also shown in Fig. 80c, is the corresponding digital simulation. The latter features a rate-limited actuator model, but the deadband effect has been omitted for clarity. Similarly, Figs. 81a,b and c show the results for the pitch controller. This is of greater interest because, like the Mk 3, the earlier pitch autopilot employed a digital compensator.

Clearly the quantization of the signal levels has not de-stabilised the control loops, and the results are generally as

predicted by the digital simulations. The shape of the transients are correct, though those for the pitch loop are masked by the more severe deadband effect. This is shown in Fig. 82, which depicts the roll and pitch attitude under steady state conditions. The extent of the oscillations compares with the digital simulations of the later autopilot in Figs. 61 and 71, which do include deadband modelling.

7.2 Flight results.

7.2.1 Summary of the flight.

The DFCS was flown for the first time on the 18th October 1983, from a Ministry of Defence range. The flight lasted 16 minutes, during which time the operator satisfied himself that the system was performing as required. The autopilot and height-lock modes were used, with recovery achieved by parachute deployment. Momentary breaks in telecommand transmission, induced at the ground station, resulted in brief periods of failsafe command. A full account of the flight is given in the references (26), so the analysis here will be confined to the performance of the controllers.

7.2.2 Observations on the flight results.

The primary observation is that the DFCS performed in a similar manner to the existing analogue flight control system. In this, the desired objective was realised. To make a qualitative comparison with the results of a previous flight would not be instructive. This is because the performance depends on many factors, including the state of trim, weight and thrust available. Stabileye no.43 was used to

test a larger, more powerful engine installation, which resulted in a heavier aircraft. Furthermore, it is standard practice when flying a new airframe, as was no.43, to use the first flight to assess the state of trim. Therefore the performance must be considered in isolation.

The results are provided by the real time record, which include both the telecommand and telemetry histories. As such, they are subject to errors in the encoding, decoding and display systems.

7.2.2.1 State of trim.

The type of control law used, and the nature of the pre-flight adjustments mean that the aircraft must be trimmed. This is especially so for roll attitude, as no provision to balance the aircraft about the OX axis is made during the pre-flight checks. The mean attitudes achieved for zero demands were as follows:

Roll	+0.0785 rad	(to port)
Pitch	+0.0140 rad	(up)

7.2.2.2 The roll step response.

The roll attitude controller performed as expected, with good demand following if the effect of the trim error is taken into account. This is illustrated by Fig. 83 which shows a sequence of negative and positive roll excursions. The first manouvre is a close approximation to a step demand, with 41.0 m/s indicated airspeed. If this is compared with the digital simulation response at 40m/s, in Fig. 61,

it is seen that the rise time, level of damping and steady state variation are as predicted.

7.2.2.3 The pitch step response.

The pitch attitude controller gave acceptable results for positive demands ("pitch-up"), in terms of speed of response, level of damping and steady state variation. This is shown in Fig. 84, which also shows a negative pitch excursion. As can be seen, the demand following for this case is poor. There are several possible reasons why this occurred. First, it may have been due, in part, to the trim condition of the aircraft, but this is unlikely to have been entirely responsible. Secondly, The controller may have been malfunctioning, but there are indications that this was not so. That is to say, the controller is capable of achieving a positive pitch attitude demand, and also, the system was checked prior to flight by inclining the aircraft on a moving platform, and observing the deflection of the control surfaces. This process had given satisfactory results for both the roll and pitch axes. The most likely explanation is that the simplified aircraft transfer function is not valid at large negative pitch attitudes.

7.2.2.4 The height-lock performance.

Fig. 85 shows the climb to height from launch. The climb rate is virtually constant at 2.5m/s until the height error is less than 20m. At this stage the loop enters its linear region, and the rate of climb reduces to zero. Once the demanded height of 300m is reached, the HTERR function indicates that the steady state variation is +/- 4.0m.

Part III DESIGN OF AN ADAPTIVE ROLL ATTITUDE CONTROLLER.

Chapter 8 Introduction to part III

Following the successful conclusion of the flight trials with the fixed gain DFCS, attention was turned to the use of different control strategies to take advantage of the flexible nature of the software. It was hinted in section 5.3 that there were no hard-and-fast guidelines to follow in designing the fixed gain autopilot, and so it is sensible to re-assess the design criteria at this stage. It is not a particularly complex control problem compared with aircraft which must accommodate a broader flight envelope, and in many respects a fixed gain autopilot provides an adequate solution. Furthermore, the level of complexity that must be included in a manned aircraft is difficult to justify for a low cost R.P.V., so the case for developing an adaptive autopilot is questionable.

There is one aspect in which the fixed gain controller fares poorly - the trim error problem apparent in the flight results. This is due to misalignment and imbalance of the airframe, and misalignment of the servoactuators, which act as a disturbance in the closed loop. They would normally be eliminated by adding corrections to the attitude commands at the ground station, or by the application of trim tabs to the airframe after an initial proving flight. Such procedures are acceptable for a research vehicle, but they are out of the question for a production R.P.V., (which should exhibit mimimum trim error without operator intervention). The solution must lie with the autopilot control law. Using an integrator in the forward path to overcome the steady state error would be unwise, as it creates a double pole at the s-plane origin and de-stabilises the system. An

attractive alternative is to reduce, not eliminate the offset, by increasing the loop gain. For significant improvement the increase should be at least twofold, which will lead to unacceptable levels of overshoot. The desired solution is a control law that will apply larger gains for controlled intervals only, so as to reduce the disturbance error and avoid excessive overshoot. If compensation is used to suppress the gain during the transient period, the order of the system will be increased, which is not generally desirable. The other option is to use a variable gain, as part of a variable structure system (V.S.S.).

The particular type of structure change investigated here is the use of a sliding mode, as developed by Emelyanov et. al. (27-37) in the 1960's. A useful introduction to the subject is provided by Utkin (35), and a review of its relation with other control strategies is given by Maslen (38). More comprehensive treatments are to be found in Itkis (36) and Utkin (37). Recent work by White (39,40) has led to a more workable design technique, and a valuable means of system design when states are unavailable.

The structure, (usually just a gain), is varied in such a way as to force the output to follow an implied model. The choice of this model is not free, in earlier work it was of one order less than the plant, but White has shown that viable systems can be designed with lower order models than this. Obviously it is selected to have some desired property. For Stabileye, a monotonic response is attractive, especially when the payload includes real-time T.V. monitoring equipment, so the model can be chosen with this in mind.

Most adaptive autopilot schemes (41-45) require more knowledge about the system state than a fixed gain design. Sliding mode V.S.S.

ideally requires all of the derivatives of error up to (n-1), where n is the order of the plant. This is acutely difficult for a low cost R.P.V. which must use the least number of sensors on the grounds of minimal cost, weight and complexity. Hence it is necessary that these derivatives shall be generated by some artificial means.

Part III of this thesis is confined to a theoretical design study only. This is because the Stabileye project ran into difficulties throughout 1984, and was eventually cancelled at the end of that year, so flight trials of a V.S.S. autopilot were not possible. The purpose of this investigation is to indicate by simulation the sort of improvements that can be achieved, but how these can be negated by the effects of the physical constraints of a real system. The roll autopilot only has been studied: This is because the pitch autopilot exhibits the same sort of difficulties, but these are compounded by its being of higher order. This is not a problem in itself, if it can be assumed that additional sensor information is available. As it stands however, the prospect of developing a variable structure pitch autopilot using measured attitude only is an unrealistic undertaking.

The hybrid simulation stage will not be performed for this work. Its most useful role is in proving the proper working of the practical implementation of the DFCS. As this has already been shown to be so, there is little point in repeating the excercise.

Chapter 9 Theory of variable structure systems.

9.1 Sliding modes.

A variable structure system is one in which the structure is changed during the transient process in accordance with the current value of the error and its derivatives. These changes can include altering gains, adding or removing feedback paths, or switching the type of compensation. Other more obscure examples are; the inclusion of a multiplicative mode, where the gain is made a multiple of the error (46), the use of a strategy of fixed control increments where the next increment is determined by the sign and magnitude of the error and error rate (47), or by switching between a type I and a type II system (48). The possibilities are endless, and they can all be achieved without specialised hardware, when using a software-based controller. Many of the methods are difficult to analyse. Having been designed (say by trial and error) for one set of plant dynamics, they could give unpredictable results as the plant changes. The use of a sliding mode avoids these problems.

If the structure is switched on a surface in state space, a sliding mode will occur if, in the vicinity of the surface, the state trajectories of the prevailing structures are directed towards this surface. Consequently once this discontinuity surface has been reached, the system will rapidly switch from one structure to another as it crosses, and re-crosses the boundary. Hence it will remain within a very small region of the surface, and will slide along it in state space until it reaches an equilibrium condition. The motion of the controlled system at this time will depend on the equation of the switching surface alone.

This is attractive for the following reasons:

The controlled response is independent of the plant and can be made insensitive to parameter changes.

If the system to be controlled is of order n, the equation of the switching surface will be of order (n-1).

The constituent structures need not be well behaved, and may even be unstable.

The concept can be illustrated by a simple example. Consider the time invariant plant whose motion is described by:

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = -a_2 x_2 - a_1 x_1 + u \qquad 9.1$$

where x_1 is the error, a_1 and a_2 are the constant parameters, and u is the control signal.

Let u be a piecewise linear function of x_1, so:

$$u = -\psi x_1 \qquad 9.2$$

where ψ can have the values α and $-\alpha$.

If plant coefficient a_2 is negative, and the value of α is chosen so that the system with $\psi = \alpha$ has complex eigenvalues, and with $\psi = -\alpha$ has real eigenvalues, the state trajectories will be as shown in Figs. 86 a and b. Both structures are unstable.

Now the system structure is switched on the surfaces described by:

$$x_1 = 0 \text{ , and } \sigma = c_1 x_1 + x_2 = 0 \qquad 9.3$$

Here, $c_1 > 0$, is constant, and is selected so that the straight line $\sigma = 0$ occupies the position shown on Fig. 86b.

If the law of structure change is:

$$\psi = \alpha \ , \quad \text{if } x_1\sigma > 0$$

$$\text{and } \psi = -\alpha \ , \quad \text{if } x_1\sigma < 0 \qquad 9.4$$

then the state trajectory will be as shown in Fig. 86c. As can be seen the trajectory reaches the surface $\sigma = 0$ from any initial condition, and then slides down the surface to the origin. The motion of the controlled system in the sliding mode is described by the equation for the switching surface, i.e.

$$c_1x_1 + x_2 = 0$$

$$\text{or } c_1x_1 + \dot{x}_1 = 0 \qquad 9.5$$

This is a first order differential equation which is stable as $c_1 > 0$, and has been produced from two unstable structures.

Notice that if the plant parameters a_1 and a_2 vary over a known range, then the stable asymptote in Fig. 86b will also lie within a known sector of the phase plane. If the switching surface, $\sigma = 0$, is chosen so that it lies between $x_2 = 0$ and the sector covering the possible stable asymptotes, then sliding will always occur. So the response of the system will be predictable over a range of plant parameter values, and once the switching surface is reached, the motion will be insensitive to changes in the plant.

9.2 Sliding mode control for n-th order plants.

The controller design can now be resolved into three tasks:

Selecting a suitable switching function.

Ensuring that a sliding regime will occur.

Ensuring that the state will approach the surface from any initial condition in state space.

The example given previously was second order so it was possible to do these intuititively by inspection of the phase plane. Higher order systems require a design method to assist in finding a suitable switching function, and the structures necessary to achieve sliding. The earlier work, as summarised by Itkis (36) revolves around finding sets of inequalities to relate the plant parameters, the equation for the switching surface and the gains to be used. As expected, these inequalities severely restrict the choice of σ. For example, consider those developed for the control of an n-th order plant with varying parameters (36).

The plant to be controlled is described by:

$$\dot{x}_i = x_{i+1} \qquad (i = 1....n-1)$$

$$\dot{x}_n = -\sum_{i=1}^{n} a_i(t)x_i + b(t)u \qquad 9.6$$

where x_1 is the error, $a_i min \leq a_i(t) \leq a_i max$, and $bmin \leq b(t) \leq bmax$ are the varying plant parameters, and u is the control.

In order to achieve insensitivity to all plant parameter changes over the bounded ranges, it is necessary that x_1 and all (n-1) derivatives of x_1 be fedback via switched gains. So the control function is given by:

$$u = -\sum_{i=1}^{n} \psi_i x_i \qquad 9.7$$

The switching function is:

$$\sigma = \sum_{i=1}^{n} c_i x_i \quad , c_n = 1 \qquad 9.8$$

and the law of structure change is:

$$\psi_i = \alpha_i \quad \text{if} \quad x_i \sigma > 0$$

$$(i = 1....n) \qquad 9.9$$

$$\beta_i \quad \text{if} \quad x_i \sigma < 0$$

For a sliding mode to occur on $\sigma = 0$ it is necessary that:

$$\lim_{\sigma \to 0} \ (\dot{\sigma}\, \sigma \leq 0) \qquad 9.10$$

Inequalities are derived that satisfy the broader condition:

$$\dot{\sigma}\, \sigma < 0 \qquad 9.11$$

These are:

$$\alpha_i \geq \max_t \frac{1}{b(t)} (c_{i-1} - a_i(t))$$

$$(i = 1....n) \qquad 9.12$$

$$\beta_i \leq \min_t \frac{1}{b(t)} (c_{i-1} - a_i(t))$$

We see that these are always satisfied if $b(t)\alpha_i$ and $b(t)\beta_i$ are made sufficiently large, (positive and negative respectively). As they satisfy eqn. 9.11 the state will reach $\sigma = 0$ for any set of initial conditions, and motion will then be stable provided the coefficients forming the switching surface equation satisfy the Routh-Hurwitz conditions. The restrictions imposed by eqn. 9.12 do not appear too stringent, but it must be considered that the control strategy described by eqns. 9.7,9.8 and 9.9 is complete.

If the strategy does not switch all of the derivatives, then extra inequalities must be satisfied to ensure sliding.

The method developed by White (39) is attractive because it provides a more immediate impression of the allowable switching functions for a given system with a given control strategy. It also gives information about the motion of the state about the switching surface, which is valuable when x_n is not available to form σ. The method is outlined as follows:

Consider the control of an n-th order plant using, first of all, fixed state feedback. The system to be controlled is as given in eqn. 9.6 but with fixed parameters. This can be expressed in companion form as:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{b}u$$

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & \cdots & 1 \\ -a_0 & -a_1 & -a_2 & \cdots & -a_{n-1} \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \qquad 9.13$$

The switching function is described by:

$$\sigma = \sum_{i=1}^{n} m_{i-1} x_i \qquad (i = 1 \ldots n) \qquad 9.14$$

This can be written in scalar product form:

$$\sigma = \mathbf{m}^t \mathbf{x} \qquad 9.15$$

where $\mathbf{m}^t = (m_0\ m_1 \ldots m_{n-1})$, $m_{n-1} = 1$

^{t} is considered as a linear operator in n-th order vector space. This has an (n-1)th order null space, and a first order range space. We have already seen in section 9.1 that in second order state space, the null space of the switching function is first order, and is given by eqn. 9.5.

The object with a fixed structure is to select a control signal

$$u = - \mathbf{K}^t \mathbf{x} \qquad 9.16$$

that will force the system onto the null space, and then remain there. This is achieved by placing (n-1) eigenvectors in the null space using state feedback design, and ensuring that the remaining eigenvector is stable and fast compared with the null space dynamics. Now, if $\mathbf{K}^t$ is chosen to place (n-1) eigenvectors in the null space, the range space dynamics can be shown to be:

$$\dot{\sigma} = \lambda_n \sigma \qquad 9.17$$

where λ_n is the eigenvalue associated with the eigenvector not placed in the null space.

If λ_n is slow or unstable, the gain vector can be augmented to improve the range space dynamics. The modified control signal can be written as:

$$u = -(\mathbf{K}^t + \Delta \mathbf{K}^t)\mathbf{x} \qquad 9.18$$

where $\mathbf{K}^t$ is as defined before.

Now the range space dynamics with this augmented control can be shown to be:

$$\dot{\sigma} = \lambda_n \sigma - \Delta \mathbf{K}^t \mathbf{x} \qquad 9.19$$

Note that if ΔK^t is chosen as:

$$\Delta K^t = \beta m^t \qquad , \beta \text{ is constant.} \qquad 9.20$$

then this gain augmentation has no effect in the null space, where $\beta\, m^t x = 0$, but the range space dynamics will be:

$$\dot{\sigma} = (\lambda_n - \beta)\sigma \qquad 9.21$$

which can be made as fast and stable as required by appropriate selection of β. This is an alternative interpretation of state feedback design, and is only given here to illustrate the relationship of V.S.S. control to that technique.

Now consider a variable structure system, which will occur if the gain augmentation vector is switched. The state will remain in the null space if:

$$\text{sgn}\ (\sigma\ \dot{\sigma}) = -1 \qquad 9.22$$

In the region of $\sigma = 0$, eqn. 9.19 reduces to:

$$\dot{\sigma} = -\Delta K^t x \qquad 9.23$$

Therefore sliding motion will occur if the law of structure change is chosen to satisfy:

$$\text{sgn}\ (\Delta K_i x_i \sigma) = 1 \qquad (i = 1 \ldots n) \qquad 9.24$$

This is full state switching as proposed in eqn. 9.9, but the inequalities in eqn. 9.12 have been avoided. This is because the relationship of the gains, the plant parameters and the switching coefficients are already defined by selecting the fixed part of the gain vector to place (n-1) eigenvectors into the null space. This can also cope with varying parameters by making the switched gain vector adequate

to compensate for changes in K^t necessary to do this as the plant, and hence the eigenvectors vary.

Eqns. 9.23 and 9.24 do not apply for all state space, so reachability conditions have to be found. These usually require that the dominant modes of the response are second order and oscillatory, but not necessarily stable, when $\Delta K_i > 0$.

9.3 Practical difficulties with the application to real systems.

Any non-linearities, whether noise or delay can be interpreted as functions of time that are added to the true switching function, causing the system to change structure at some point other than the correct instant. Hence, a "measured" switching point can be envisaged as part of a certain variable surface, passing through the origin, which is constrained to move in some neighbourhood of the ideal switching plane. (For second order state space this measured surface will occupy a sector about the straight line, $\sigma = 0$). Itkis (36) develops conditions which ensure stability in these quasi-sliding modes, and make them similar in their action to motion in a sliding mode proper. As a result the advantages of V.S.S. are preserved in a real system.

The specific problems treated by Itkis will only be described here in terms of their effect on this particular application. These effects will be illustrated in Chapter 10 by the digital simulation results. The one aspect that Itkis does not cater for, that is the design of the system with a reduced order switching function, will be investigated by examining the range space dynamics as shown by White (39).

9.3.1 Non-infinite switching rate, and limited gains.

To remain within an infinitely small region of the switching surface the system must be capable of changing structures at an infinite frequency. This is not possible, and in practice the state spends a finite time either side of the null space. Hence it will penetrate each region before returning, thus performing periodic oscillations about $\sigma = 0$. These will decay as the error decreases. The implication for a sampled data system is clear. The magnitude of these oscillations will depend on the sampling rate and the size of $\dot{\sigma}$. The latter is proportional to the switched gain vector when in the region of the switching surface, so the combination of high gains and low sampling rate is to be avoided. This combination could eventually lead to the state passing through a switching region between sampling instants, and not changing structure. This will give an unpredictable response and even instability if an unstable structure is allowed to persist. For this application, the maximum switching rate will be constrained by the servomotor update rate (40.0Hz), and also by the rate limit in the actuator itself.

There are other reasons for limiting the switched gain component. Eqn. 9.19 suggests that faster range space dynamics can be achieved by increasing gains. Furthermore, the inequalities in eqn. 9.12 can always be met with an adequately large gain. This is not wise as any real system will be represented by a model which ignores high frequency roots. As the gain, and hence the bandwidth, is increased these will become significant, so that the model is no longer valid.

For the DFCS, the position limit of the actuator presents a physical restriction on the gain. This means that any scheme which relies on

high gains to maintain stability must be discounted. However it also provides a method of limiting the gain during the transient period only. This fortuitous condition is an advantage when the object is to reduce the steady state trim error, but it should only be exploited with extreme caution.

9.3.2 Unavailable states.

The requirement that all of the derivatives of error up to x_n be available, imposes serious difficulties. One solution would be to use dedicated sensors. This is only practical for systems which can be represented by low-order models. Stabileye, like many other control design specifications, is required to use the minimum number of sensors, so computed approximations must be used. The literature cites the use of continuous filters (36,29) and continuous differences (30) to simulate differentiation. Clearly this application would use a sampled data method. These all introduce elements of noise and delay which degrade the quality of the control. By reducing the delay (or phase lag) the approximation to pure differentiation is improved, but the accuracy of the measuring device must be increased. The compromise reached will be dictated by the capabilities of the A/D converter.

The problem of unavailable states falls into two categories, namely states missing from the switching structure, and states missing from the formulation of σ. These will be discussed separately.

9.3.2.1 States missing from the switching structure.

By switching all of the higher derivatives according to eqn. 9.24

(or 9.9) the system will suffer from reduced noise immunity. Hence it may be desirable to switch only $x_1 - x_k$, $(1 \leq k \leq n-1)$ even if the states are available. This will reduce the choice of switching surfaces, and total parameter insensitivity will be lost. The choice of σ will be further reduced if the state is not fedback via a fixed gain path either. This is analogous to state feedback design, where complete freedom of choice of controlled performance is only possible if all the states are available. It has been shown (40) also that constraints are imposed upon the constituent structures (and hence the gains), to ensure reachability with reduced switching vectors. These constraints are avoided if the range space dynamics are stable.

9.3.2.2 States missing from the formulation of σ.

To consider the effect of omitting x_n from the formulation of σ, it is necessary to examine the dynamic behaviour of σ about the null space. For the full state case the range space dynamics are given by:

$$\dot{\sigma} = \lambda_n \sigma - \Delta K^t x \qquad \text{9.19 repeated}$$

If it is assumed that for small changes $\dot{\sigma} = 0$, $\Delta K^t x$ can be considered to be fixed, then the trajectory can be approximated by a straight line, passing through:

$$\dot{\sigma} = 0 \ , \quad \sigma = \frac{\Delta K^t x}{\lambda_n} \qquad 9.25$$

From eqn. 9.24 the signs of $\Delta K^t x$ and σ are the same, so the line will be as shown in Fig. 87 a,b for stable and unstable λ_n respectively. (From Fig. 87b it can be seen that the stationary point must be larger

than σ for the unstable λ_n.) The switching structure is arranged so that as σ goes through zero, the gain vector changes sign, so the stationary point switches to the opposite half of the $\dot{\sigma} = 0$ axis. The trajectory now follows a new straight line, as shown in Fig. 88. This new locus returns the state to the $\sigma > 0$ region to repeat the process, but the stationary point will be nearer the origin as the magnitude of $\mathbf{x}$ will have reduced. The complete action can be summarised as follows. On reaching the switching surface, the trajectory jumps rapidly between two limits either side of $\dot{\sigma} = 0$, which shrink toward the origin. Theoretically the trajectories do not penetrate into the range space, though in practice this is not possible.

This conctruction can also be applied to the reduced σ case. If x_n is not included in the formulation of σ, the linear operator $\mathbf{m}^t$ will be :

$$\mathbf{m}^t = (m_0\ m_1\ m_2\ \ldots\ m_{n-2}\ 1\ 0) \qquad 9.26$$

The null space is constrained to include the x_n axis. $\mathbf{K}^t$ is chosen to provide the desired eigenvectors, but only (n-2) may lie in the null space, so the range space dynamics become second order. $\dot{\sigma}$ is no longer proportional to σ, and does not include a term in $\Delta\mathbf{K}^t$. Hence the existence conditions for sliding motion, (eqn. 9.22), cannot be satisfied. However quasi-sliding conditions can be achieved, which approximate to a full state sliding motion. It can be shown that: (34)

$$\ddot{\sigma} = (\lambda_n + \lambda_{n-1})\dot{\sigma} - (\lambda_n\lambda_{n-1})\sigma - \Delta\mathbf{K}^t\mathbf{x} \qquad 9.27$$

Using the same approximation as before, the stationary point can be

given by:

$$\dot{\sigma} = 0 \quad , \quad \sigma = \frac{-\Delta K^t x}{\lambda_n \lambda_{n-1}} \qquad 9.28$$

This switches position as the sign of σ changes, causing the sign of $\Delta K^t x$ to change. The approximate trajectories will not be straight lines as for the full state case. There are now several possibilities for the shapes of the loci depending on λ_n and λ_{n-1}, which can be real, complex, stable or unstable. The only combination that will not converge on to the switching surface is when both roots are real and unstable.

An example is given in Fig. 89 of the range space trajectories associated with real, stable eigenvalues. Significant excursions either side of $\sigma = 0$ now occur. The extent of these depends on the magnitude of $\Delta K^t x$, which must be limited if the oscillation is not to be apparent at the output. This, in turn, limits the range of parameter variation that a reduced σ system can cope with. The quality of control is less than that of the full state case, but this is still a valuable result in its own right. It provides a design method for the very likely situation where the highest derivative of error is not available, and cannot be sensibly estimated.

9.3.3 Forced motion.

All of the foregoing refers to the free motion of the system, and is attempting to force the error states to zero. Many control systems have a reference input that is to be followed, and this can cause problems for a sliding mode controller. Essentially the forcing function , or a disturbance, will push the trajectory off the switching surface

as the steady state is approached. The literature suggests methods to overcome this. These include the use of switched actuator feedback (37), which strives to make the actuator appear as an integrator, or the addition of a fixed amplitude relay signal (49). In the presence of a reference input the following general control strategy is proposed:

$$u = -(K^t_1 + \Delta K^t_1)x + (K^t_2 + \Delta K^t_2)r + y(\mathrm{sgn}(\sigma)) \qquad 9.29$$

where r is the reference input expressed as a vector, and $y(\mathrm{sgn}(\sigma))$ is a relay component.

It has been shown (50) that for certain combinations of input and system type this reduces to standard unity feedback of the form:

$$u = K^t(r - x) + \Delta K^t(r - x) \qquad 9.30$$

This simplification can be made if the system would exhibit zero steady state error using only the fixed gain vector. The aircraft transfer function is type 1 and the reference input is a step, so a control signal of the form shown in eqn. 9.30 will maintain sliding as the steady state is approached.

Chapter 10 Use of a V.S.S. to reduce the trim error in the roll controller.

10.1 Nature of the disturbance.

The flight results show a roll attitude offset of + 0.0785 rad. This is typical of an airframe that has not been trimmed, and is caused by misalignment of the flying surfaces. These can be assumed to be fixed, so the disturbance can be represented by a constant offset added to the servo-actuator output. For comparison purposes, the size of this input will be calculated to give a 0.1 rad. attitude error at steady state for the fixed gain autipilot. A type 1 system with unity feedback will have a disturbance error of:

$$x_1(ss) = \frac{f}{K} \qquad 10.1$$

where f is the disturbance, K is the forward gain.

As the fixed gain autopilot's value of K was 0.4, the simulated disturbance input will be 0.04 rad.

10.2 Design of a V.S.S. roll controller with a full state switching function.

Initially it will be assumed that:

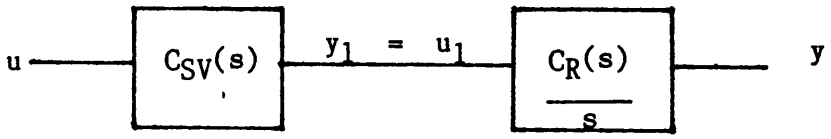
all of the states are available to form σ.

there is no disturbance.

the linearised, first order servo-actuator model is adequate.

As for the fixed gain autopilot design in Chapter 5, the digital simulation will be used to confirm the step responses. In order to

simplify the analysis the sampled nature of the control signal will be ignored, (its effect is included in the simulation) so the block diagram for the plant will be:



Control signal. Actuator position. Roll attitude.

Where $C_{SV}(s) = \frac{20.0}{s + 20.0}$, $\frac{C_R(s)}{s} = \frac{K_R}{s(s + a)}$

and K_R and a vary according to the data in Table 4. (The negative sign associated with K_R in Table 4 is to be included in the equation for the control signal.

This can be described as:

$$s(s + a)y = K_R u_1$$

$$u_1 = y_1 \qquad 10.2$$

$$(s + 20)y_1 = 20u$$

Expressing this in state variable form, with x_1 as the output:

$$x_1 = y$$

$$\dot{x}_1 = x_2 \qquad 10.3$$

$$\dot{x}_2 = x_3$$

$$\therefore \quad \dot{x}_3 = -(20 + a)x_3 - (20a)x_2 + 20K_R u$$

This is a reference following system, so the control signal should

be of the form:

$$u = K^t(r - x) + \Delta K^t(r - x) \qquad 10.4$$

where $r^t = (r\ \dot{r}\ \ddot{r})$, $\dot{r} = \ddot{r} = 0$

Without loss of generality the step response can be studied with $r = 0$ and non-zero initial conditions for x_1, so that:

$$u = -(K^t + \Delta K^t)x \qquad 10.5$$

The option to include all of the states in the switching structure will not be taken. This is because it reduces the noise immunity, and as the practical implementation of the controller must use artificially generated rate and acceleration, the states will be "noisy". Therefore only feedback of the output state will be used, so:

$$u = -(K + \Delta K)x_1 \qquad 10.6$$

The plant parameters for the 40.0 m/s case were selected, and using the results from the z-plane root locus, (presented in Chapter 5), it is seen that the gain:

$$K = 0.3 \qquad 10.7$$

gives the following closed loop pole positions:

$$z_1 = 0.917, \quad z_2 = 0.7935, \quad z_3 = 0.512 \qquad 10.8$$

The equivalent s-plane roots of the characteristic equation are:

$$(s + 3.698)(s + 9.252)(s + 26.78) = 0 \qquad 10.9$$

The fixed gain vector in eqn. 10.7 places (n-1) eigenvectors into the null space if m^t is chosen as:

$$m^t = (34.21 \quad 12.95 \quad 1.0) \qquad 10.10$$

The range space eigenvalue is stable and fast compared with the null space, which will have a response of the type required. As the range space eigenvalue is stable, there are no special reachability conditions to be applied to the choice of ΔK.

Fig. 90 compares the step responses at 40.0 m/s for this system with $u = -(0.3 \pm 1.0)x_1$ and $u = -(0.3)x_1$. (The latter is the fixed state feedback control necessary to force the state onto the null space with a fast range space eigenvalue.) This shows that once the switching surface is reached the state does follow the dynamics of the surface exactly. Note that the switched scheme is faster because it reaches $\sigma = 0$ before the fixed version. Fig. 91 shows the time response of σ for this system, which does not remain within a small region of $\sigma = 0$ in the sliding mode. The effect of the sampled nature of the control signal is plainly evident. The structures do not change at the instant the switching surface is reached, but instead they change at fixed intervals, 0.025s apart. The oscillations produced are too fast to be visible at the output, and ultimately decay to zero. The overall performance has the appearance of true sliding motion.

Fig. 92 shows the step responses at 22.0 and 50.0 m/s. These can be compared with the fixed gain autopilot results in Chapter 5, ($u = -(0.4)x_1$), reproduced in Fig. 93. The reduction in sensitivity to plant parameter changes is considerable, but it is clear that total invariance has not been achieved. Apart from the different lengths of time each system takes to reach the surface, motion in the quasi-

sliding mode is also different. This is for two reasons. The non-infinite switching rate means that neither system state remains within a small region of $\sigma = 0$. This was not troublesome for the 40 m/s case, but it results in a mean offset to one side of the surface for the 22 and 50 m/s transients. Secondly the switched gain vector is inadequate to place (n-1) eigenvectors into the null space of m^t (given by eqn. 10.10) as both K_R and a vary. To do this a complete switching structure would be necessary, feeding back the error rate via a switched gain as well.

10.21 Introducing the disturbance.

The effect of the disturbance on the full state σ system is shown by examining the range space. The equations describing the system in eqn. 10.3 are modified to include:

$$u_1 = f + y_1 \qquad 10.11$$

so:

$$\dot{x}_3 = -(20 + a)x_3 - (20a)x_2 + 20K_R u + 20K_R f + K_R \dot{f} \qquad 10.12$$

as f is a fixed disturbance, $\dot{f} = 0$.

For a full state switching function with output feedback only:

$$\dot{\sigma} = m_1 \dot{x}_2 + m_2 \dot{x}_3 + \dot{x}_3$$

$$= -20K_R K x_1 + (m_1 - 20a)x_2 + (m_2 - (20 + a))x_3 + 20K_R f - 20K_R \Delta K x_1 \qquad 10.13$$

Now K has been chosen to satisfy the following:

$$s^3 + (20 + a)s^2 + 20as + 20K_R K = (s^2 + m_2 s + m_1)(s - \lambda_3) \qquad 10.14$$

and by equating coefficients in the above it can be shown that eqn. 10.13 reduces to:

$$\dot{\sigma} = \lambda_3\sigma - 20K_R\Delta K x_1 + 20K_R f \qquad 10.15$$

Using the straight line approximations given in section 9.3.2.2, the the trajectories will be as shown in Fig. 94. Sliding will break down when the state trajectory no longer crosses the the $\sigma = 0$ axis. As a first approximation this occurs when one of the offset stationary points coincides with the origin., i.e. when:

$$\dot{\sigma} = 0 \ , \quad \frac{20K_R(\Delta K x_1 - f) = 0}{\lambda_3} \qquad 10.16$$

$$\text{so } x_1 = \frac{f}{\Delta K}$$

Equation 10.16 and Fig. 94 reveal that, for positive or negative disturbance, this will always occur when:

$$\text{sgn}\,(x_1\sigma) = 1 \ , \quad \text{so } \Delta K > 0 \qquad 10.17$$

Provided the system is stable with a fixed gain of $(K + \Delta K)$, and does not re-cross a switching boundary, the steady state value will be:

$$x_1(ss) = \frac{f}{K + \Delta K} \qquad 10.18$$

which shows an improvement on the fixed gain case in eqn. 10.1. Hence the desired mechanism in the presence of a disturbance is for the sliding mode to modulate the gain during the transient, thus prev-

enting overshoot. As the steady state is approached, sliding should break down to the structure corresponding to the large positive gain.

The simulation results for the full state system with a disturbance are shown in Figs. 95 and 96. They are not as predicted by the analysis. The variable structure system achieves a similar steady state error as would a fixed gain controller with the same value of K. The reason for this is evident from Fig. 96, which shows that the switching function enters a limit cycle before the point at which sliding breaks down. The limit cycle is caused by the fixed interval between structure changes which, when combined with the disturbance offset, prevents the range space oscillations from decaying to zero. These oscillations are too fast to be apparent at the output, which settles at an indeterminate value.

It is possible that increasing the sampling rate will allow the system to behave as expected, resulting in a breakdown in the sliding mode. However, the 40.0 Hz servomotor update rate represents a physical limit that cannot be exceeded, and so this would be an un-realistic course of action. It is suggested that a strategy that relies on rapid structure switching is unsuitable for this application, hence full state switching function schemes will be abandoned in favour of reduced order switching.

10.3 Design of a V.S.S. roll controller with a reduced state switching function.

This is more attractive than the former for a number of reasons. It does not require that the second derivative of error be available, which makes practical implementation more realistic. The rate of structure switching is much slower, and less likely to be impeded

by the sampling rate of the control system. Furthermore, a gain K exists that places (n-2) eigenvectors in the null space, for the 22, 40 and 50 m/s dynamics. (This was not so for the second order null space of the previous example.) Unfortunately the large values of ΔK desirable to reduce the steady state disturbance error, will almost certainly give rise to visible ripple at the output.

As before the fixed gain K is chosen as 0.3, but the switching function is reduced to :

$$m^t = (3.698,\ 1.0,\ 0.0) \qquad 10.19$$

The range space eigenvalues are both stable (see eqn. 10.9), but one of them is not fast compared with the null space. This is especially so at 22 m/s where λ_3, λ_2 would be -21.7 and -5.178. This is only just faster than the null space, but in practice proves to be adequate. After experimentation the switched gain component was reduced from that value used in the full state example, as it lead to excessive output ripple.

Fig. 97 compares the responses at 40 m/s with $u = -(0.3 \pm 0.6)x_1$ and $u = -(0.484x_1 + 0.05096x_2 + 1.3377\ 10^{-3}x_3)$, which is the fixed state feedback necessary to give the desired null space with fast, stable range space dynamics ($\lambda_3 = \lambda_2 = -20.0$). The latter represents motion according to the null space equation, once the range space contributions have decayed, (say after t = 0.25s). The switched system follows the null space well, but the expected ripple is evident, though not obtrusive. Fig. 98 shows the time response of σ, with damped oscillation and a much slower switching rate than the full state case. The reduced sensitivity to plant parameter changes is shown in Fig.99, which gives the responses at 22 and 50 m/s. It is seen that once the two transients have reached the switching surface they follow the null

space well. It is only dissimilar range space dynamics, evident by the different frequency and amplitude of the ripple, that distinguishes them.

10.31 Introducing the disturbance.

The effect of the disturbance on the range space is similar to the full state case. For the reduced switching function:

$$\ddot{\sigma} = m_1\dot{x}_3 + \ddot{x}_3 \qquad 10.20$$

$$= -20K_RKx_1 - 20ax_2 + (m_1 - (20 + a))x_3 + 20K_Rf - 20K_R\Delta Kx_1$$

K has been shown to satisfy:

$$s^3 + (20 + a)s^2 + 20as + 20K_RK = (s + m_1)(s - \lambda_2)(s - \lambda_3) \qquad 10.21$$

and so it can be shown that eqn. 10.20 reduces to:

$$\ddot{\sigma} = (\lambda_2 + \lambda_3)\dot{\sigma} - (\lambda_2\lambda_3)\sigma - 20K_R\Delta Kx_1 + 20K_Rf \qquad 10.22$$

The state trajectories will be as shown in Fig. 89, but with an offset in the positions of the stationary points. As before, sliding will break down when the trajectory no longer crosses the $\sigma = 0$ axis. The analysis is equivalent to that in section 10.2.1.

The simulation results for the reduced state switching function in the presence of a disturbance are given in Figs. 100 and 101. The latter shows that, from t = 1.0s onwards, σ does not change sign and settles at a positive steady state value. The final value of x_1 is 0.044 rad, which is 44% of the offset that would be given by the fixed gain scheme. This could be reduced further if a greater amount of ripple could be tolerated during the transient period. The point at which sliding breaks down is also predicted by the analysis.

10.4 The effects of the non-linear servo-actuator.

One of the advantages of using the digital simulation is the ability to examine the effect of the non-linear servo-actuator. It is found that the sensitivity reduction achieved in the previous section is overwhelmed when a realistic model is used. This particular application's objective of reducing trim error, which could be regarded as a by-product of V.S.S, is not effected. Section 10.2 concluded that the very rapid full state switching was unsuitable due to the sampling effect, so these results have been confined to the reduced state configuration.

Fig. 102 shows the step responses for 22 and 50 m/s with no disturbance input. The dominating feature is the constant rate period in the slow response, corresponding to a fixed (limited) actuator position. This does occur in the 50 m/s transient, but to a lesser extent. If these responses are compared with those for the fixed gain autopilot ($u = -0.4x_1$), also shown in Fig. 102, it is seen that the V.S.S. still shows some improvement. However the difference is so small that the reduction in plant parameter sensitivity could no longer be justified as a reason for implementing a V.S.S. control law. The only significant quality remaining is that the faster V.S.S. transient is monotonic, while the fixed gain controller response is not.

All three aspects of actuator non-linearity are seen in Fig. 103, while the effect on the switching function time response is shown in Fig. 104. This should be compared with Fig. 98. Apart from the position limiting effect, where both the output and switching function exhibit a constant rate, there is also a period where the rate limit impairs the range space dynamics. The problem extends beyond the negation of the improvement in sensitivity reduction, it is possible there are

sets of initial conditions that will lead to instability when the movement of the controlling member is limited. This has been shown (51) for a situation where a variable structure control system is used to control an otherwise unstable plant ($a < 0$). As this roll system is third order and features rate as well as position limiting, the graphical techniques of (51) cannot be used here. Empirical solution using digital or hybrid simulation may be the only satisfactory method of indicating system stability. (That is with the simplified aircraft transfer functions, whereas actual system stability could only be shown with a flight trial.)

Fig. 105 shows the step responses for the 22 and 50 m/s cases in the presence of a disturbance. The steady state error is similar to that achieved with the linear servo-actuator model, and therefore much better than the fixed gain result. This is because the actuator tends towards the linear model as the steady state is approached.

It was stated in section 9.3.1 that the actuator limits provide an artificial means of reducing the gain during the transient period, and that this might be exploited. These results show less ripple than those for the linear actuator, so it is possible that ΔK could be increased. This was tried using $u = -(0.3 \pm 1.0)x_1$ with no ill-efects, and the expected improvement in the trim error. As there is some doubt regarding the stability of the system with a limited actuator, this approach would be unpredictable, and would require practical trials confirmation.

10.5 The effect of estimating the rate.

The z-transfer function used to model the roll rate in the simulation is an estimate itself. As it is sampled at a much higher rate (320.0Hz)

than the plant dynamics, and uses double precision arithmetic, it can assumed to represent the true rate fairly well. If the rate is not available it can be estimated from the error data. The method chosen was a finite impulse response filter of the form: (24)

$$\text{Rate}(n) = \frac{1}{T}(a_0\text{error}(n) + a_1\text{error}(n-1).....) \qquad 10.23$$

to simulate the derivative. If the number of terms in this algorithm is too great, its bandwidth will be excessive and the system noise will be amplified. It is therefore attractive to use the minimum number of terms that are necessary. The bandwidth is also related to the sampling rate, which for simplicity, remains the same as the servo-motor update rate . After experimentation, a three-point algorithm of the form: (24)

$$\text{Rate}(n) = 40.0*(1.5*\text{ERROR}(n) - 2.0*\text{ERROR}(n-1) + 0.5*\text{ERROR}(n-2)) \qquad 10.24$$

was found to be adequate. This was made more representative by quantising the error measurement to the equivalent of a 12 bit A/D converter, thus adding noise to the signal.

Fig. 106 compares the switching function formed using the z-transform model of the"true"rate, sampled at 320.0 Hz, with that formed with the"estimated" rate. There is very little difference throughout most of the transient, with the latter showing a very small delay, and a magnitude increase of approximately 20%. For this period there are no perceptible differences in the instants when the switching function changes sign. As the steady state is approached the quantisation noise becomes more significant, causing the structure to change at the wrong times. This does not effect the output response, Fig.107, which is indistinguishable from that formed with the"true"rate. This similarity extends to the step responses at all three airspeeds, with

or without a disturbance, and using an ideal or non-ideal actuator model. Figs. 108 and 109 show the output and switching function responses for the 40 m/s case with a non-linear actuator model and a disturbance input. The significance of these is that the simulated controller is realisable, and so the reduction in trim error could be achieved by the DFCS with the measured roll attitude only.

The ease of estimating the rate does not suggest that a full state switching function using estimated acceleration could be used. (Nor that the pitch controller, which would require acceleration for a reduced order switching function, is viable.) Estimated rate and acceleration using three point algorithms were tried with the roll controller but met with little success. The albeit limited switching rate of the full order switching function in section 10.2, could not be matched, and in fact the switching function was reminiscent of the reduced order example in section 10.3. The 40.0 Hz sampling rate is inadequate for the bandwidth of the acceleration signal.

Chapter 11 Conclusions.

11.1 Conclusions on the practical implementation of the DFCS.

The development of the DFCS was hindered to an extent by the smallness of the R.P.V. Research dept. at British Aerospace. Stabileye was always a low budget project, not to be compared with far bigger enterprises such as NASA's Himat (52) or any of the Teledyne Ryan R.P.V.'s (53). This was manifest in the early years by the lack of established information regarding an autopilot specification, or even the transfer functions to be used. Through no lack of professionalism, more a lack of personnel and opportunity, the development of the flight control systems has been dictated by a "fly it and see" policy. This goes some way to explaining the paucity of flight data included in this thesis, as the quality of the data-logging equipment was not what might be expected in a controlled experimental environment. Attempting to perform flight step responses was difficult; the allocated area for the flight trials is small, and the operator was unable to input, and maintain, a 0.5 rad. bank command without having to break-off to prevent going beyond a safety boundary. Therefore it should be stressed that the greater part of results taken during the flight trials revolve around the operator's observations rather than the telemetry record. In this case the primary observation was that the DFCS performed in the same manner as the existing analogue controller, which was the objective. It was also noted that the DFCS was easier to set-up.

Another important historical point is the requirement to perform flight trials at a Ministry of Defence range. This made them expensive, and few and far between, so that flight testing of fee-

- paying customers' payloads had to take precedence. This led to the cancellation of a flight of the DFCS in the Mk 1 airframe at a much earlier stage of the project than the eventual date. In fact all of the installation and control design work was originally carried for the Mk 1 airframe, which caused some delay when it had to be repeated later.

Specific conclusions about the practical implementation of the DFCS are as follows:

The use of the microprocessor-based flight control system has been shown to be feasible for a mini-R.P.V. It is comparable in terms of size, cost and weight with the analogue controller, with the exception of the battery supply, which is much larger. Any production version would include an engine-driven alternator to extend the range anyway, so this is not a real issue. This conclusion has been overtaken by the events in the electronics industry over the recent years. The microprocessor has achieved a dominant position in all process control, and the autopilot is no exception. This is particularly so for R.P.V.'s where the ability to store flight programmes is invaluable if combined with some form of inertial navigation system.

The hardware and software have been shown to operate correctly, and so form a basis for subsequent development. Two factors indicate how this development should procede. Firstly the percentage usage of the c.p.u. time was between 30-35% suggesting that the software was under-utilised. Secondly the advances in V.L.S.I. design, especially in the field of c.p.u. peripheral devices, make it possible to develop a more flexible system occupying less volume. It can be concluded that the DFCS does not take enough advantage

of the programmable system at its core. This is because at the hardware/software definition stage it was not possible to determine accurately what the percentage c.p.u. time usage would be. It was necessary to design to conservative estimates to ensure that there was enough time available to perform the required tasks.

Another factor contributing to the over-reliance on the hardware was the requirement to use existing, separate, telecommand decoding and telemetry encoding systems. This has led to a bulky, over-complex system with too many inter-connections in the aircraft loom. As most of the signals required for telemetry exist in software as part of the control process, it is sensible to compile the telemetry frame by software also. Similarly, as the interpretation of the telecommand data is performed by the telecommand interrupt routine, it is a short step to perform the byte-recognition and manipulative techniques used in the dedicated receiver/decoder. This would lead to system where interfacing to the telecommand receiver and telemetry encoder would be at the serial data level. The advantages in terms of reducing the complexity, and so improving the reliability are obvious, but there is also the attraction of being able to re-configure these structures for different formats.

It is suggested that the system should tend towards a single board computer with mostly re-configurable V.L.S.I. devices. The system should then be defined by the software, treating all of the flight control system tasks as an integrated whole. This precludes the use of an analogue back-up as configured here. Again, for a production vehicle, the inclusion of a pilot-mode-only back-up would be of limited operational use anyway, so is omissiom would not be critical.

A development of the DFCS using the improvements described here has been produced in subsequent work at Britsh Aerospace. A full description is given in a report (54), which provides an alternative follow-on to parts I and II of this thesis.

11.2 Conclusions on the fixed and variable structure control laws.

The original fixed gain autopilot objectives; to repeat the simulated step responses of the analogue autopilot design, were shown to be unrealistic due to the inadequacy of previous servo-actuator models. A new, more representative model was produced, featuring measured values of position and rate limiting. The digital simulation was adapted, and then used to confirm the responses in the presence of this non-linear element. The hybrid simulation results, (only performed for the Mk 1 airframe controller, as its main purpose of de-bugging the hardware and software was unnecessary for subsequent systems), corresponded to those of the digital simulation, proving the new actuator model. However the trim error and the apparent incorrectness of the pitch transfer function for negative attitudes, as shown by the flight results, illustrated the hazards of exclusive use of simulations rather than trials. The aforementioned difficulties in obtaining flight data left little option in this respect.

The successful flight of the DFCS with a fixed gain autopilot represented the completion of the requirements for the R.P.V. Research dept., British Aerospace P.L.C. Attention was turned to an investigation of sliding mode control. Motion in a sliding mode promises total parameter insensitivity, which leads to ideas

of a "Universal autopilot", to be applied to a range of R.P.V.'s without modification. The literature on adaptive autopilot schemes for manned aircraft (41-44), suggests that this is neither achievable or desirable. The pilot, who represents an extremely capable adaptive controller, does not require the aircraft to behave identically across the flight envelope. The position for R.P.V.'s is somewhat different. There are roles where the use of V.S.S. laws could be useful to R.P.V. control. The trim error reduction described here is one, especially if the desired motion is a very slow response. (This could be attractive for a "reconnaissance" mode where rapid movement of the T.V. camera platform can be confusing.) The fixed structure necessary to achieve a slow response would require a low gain, that would lead to excessive disturbance errors. However a V.S.S. controller could use larger gains but switch about a surface whose equation corresponds to the required damped motion. Another attractive usage would be to convert the third order pitch dynamics into a more acceptable second order response. Clearly any reduction in sensitivity to plant parameter changes would be welcome. However it is not necessarily true that the "cost" of implementing a V.S.S. controller would be justifiable. This "cost" is the requirement for additional sensors with the inherent reduction in system reliability that increased complexity brings about. Also to be considered is the extra effort required to design and prove the new controller. Design time is a significant expenditure in low volume projects such as this.

So the prospect of the "Universal autopilot" is not sensible because it requires that all of the system states be available, and also that the bounded ranges of the plant parameters be known. In short, it would need to know more about the plant rather than

less. Even if all the sensors could be provided, or the derivatives simulated, it is not guaranteed that a sliding mode would be formed. This is because the larger gains encountered in a V.S.S. tend to invalidate the linearised plant model, and the infinitely fast switching rate for true sliding cannot be achieved. Applying these points to the specific case of an R.P.V. is not encouraging. It is likely that the linearised transfer functions will not be completely representative, as time spent in wind tunnels, again, is costly for such a project. What is more, large actuator excursions induce high levels of drag, which would alter the performance from that of the model, even if it was accurate. The problem of the switching rate relates to the actuator and the controller. Most aircraft actuators will have a discrete-time input, as noise considerations usually prevent the possibility of continuous analogue driving signals. This, and the inevitably constrained rate will place an upper limit on the structure switching rate. To increase the limit would require faster servo-actuators. Penalties would have to be paid in terms of increased size, weight, cost and power consumption for each item, all serious for an aircraft application. There is also a potential problem for microprocessor-based controllers. While their flexibility ensures that the variable gains are simply implemented, they are inherently sequential. This means that they also impose a limit on the structure switching rate, but its likely to be adequate for all but the most stringent applications.

Turning to the specific conclusions to be drawn from the results in Chapter 10, there are three main points. Firstly a V.S.S. controller might be successful in reducing the roll trim

error experienced by the fixed design, without additional sensors. The simulation shows that this could be done, but there is a clear requirement for confirmatory flight trials. These would be to establish whether the doubts stated in the previous paragraph are justified, and to determine the stability of the system with arbitrary initial conditions. The extent to which the switched gain component , ΔK, could be increased, and hence the trim error reduced, also remains to be shown by a practical trial. If the simulation results can be taken as representative, then an improvement of greater than 50 % can be expected. The breakdown in the sliding mode is important here, as it implies a stationary steady state. The methods suggested by the literature for eliminating, not merely reducing, disturbance error involve a 'dynamic steady state. That is to say that both the switched actuator feedback and the use of a relay component result in a limit cycle, which would rapidly damage the actuator.

Secondly the use of a reduced order switching function has been shown to be more effective than the full state case. This is because it does not rely on very rapid structure switching and is not so perturbed by the sampled nature of the control signal. Also, as it features a lower order null space, it reqires a less complicated switched gain structure to follow this space as the plant parameters vary. Its most obvious advantage is that it does not require that the highest derivative be available. The penalty paid for these relaxed conditions is the presence of ripple at the output.

Finally, despite the lack of success in this particular application, the reduction in sensitivity to plant parameter changes is appreciable under the right conditions. These are:
The availability of the system states, though artificial generation

and reduced order switching functions go some way to relax this requirement.

The existence of an adequate actuator that complies with a linear model at frequencies up to and beyond those of the range space dynamics. If it receives its input information at discrete intervals, the update rate should be high enough to avoid disrupting the proper working of the quasi-sliding mode.

The plant should be of a type that is faithfully represented by a low order model, even when included in a high gain closed loop.

In general, mini R.P.V.'s with low cost requirements for actuators and sensors do not satisfy these conditions.

TABLES.

Table 1

The major integrated circuits used in the DFCS.

Unless otherwise stated, the designation and manufacturer of the device corresponds to the following key:

74LS... Low power, Schottky T.T.L. (Texas Inst. Inc.)

74C... CMOS with T.T.L. pin-outs, (National Semiconductor)

CD4... CMOS, (National Semiconductor)

AD... Analogue/digital hybrid, (Analog Devices)

LM... Linear, (National Semiconductor)

No.	Device	Description
I.C.1	CD4049	Hex Buffer (inverter)
2	N555	Timer
3	CD4047	Retriggerable monostable/astable
4	74LS32	Quad, 2 input positive NOR
5	74LS30	8 input, positive NAND
6	"	
7	74LS04	Hex inverter
8	74LS138	3 to 8 line decoder
9	"	
10	74LS02	Quad, 2 input, positive NOR
11	"	
12	74LS00	Quad, 2 input, positive NAND
13	74C374	Octal, D-type, flip-flop
14	"	

Table 1 cont.

No.	Device	Description
15	CD4029	Presettable up/down counter
16	"	
17	"	
18	CD4078	8 input NOR/OR
19	CD4013	Dual, D-type, flip-flop
20	CD4724	8 bit, addressable latch
21	CD4050	Hex buffer (non-inverter)
22	TL084	Quad operational amplifier, with output
		protection, (Texas Inst.)
23	"	
24	CD4042	Quad, clocked, D-type, latch
25	CD4051	Analogue multiplexor
26	74C374	Octal, D-type, flip-flop
27	"	
28	ADC80AGZ	12 bit, 25us, Analogue to Digital conv-
		erter, (Burr Brown)
29	74LS04	Hex inverter
30	74C374	Octal, D-type, flip-flop
31	"	
32	CD4013	Dual, D-type, flip-flop
33	LM224	Quad operational amplifier
34	Ad7524	8 bit, buffered, multiplying DAC
35	"	
36	"	
37	"	

Table 1 cont.

No.	Device	Description
38	AD7524	8 bit, buffered, multiplying DAC
39	"	
40	LM224	Quad operational amplifier
41	"	
42	74C373	Octal, D-type, latch
43	74LS157	Quad, 2 to 1 line data selector
44	74LS244	Octal buffer
45	CD4049	Hex buffer (inverter)
46	CD4075	Triple, 3 input OR
47	74C373	Octal, D-type, latch
48	"	
49	LM295	Ultra reliable power transistor
50	"	
51	"	
52	LM339	Quad, voltage comparator
53	CD4025	Triple, 3 input NOR

Table 2

Definition of bits in the microprocessor I/O field.

These are all managed by the TMS 9901 Programmable Systems Interface device (6). Each bit has two functions, depending on whether the processor is performing a C.R.U read, or write instruction. The base address for the TMS 9901 is °0080°.

C.R.U READ.

I/O bit	9901 bit def'n	Function
°80°	Control bit	When set, 9901 is in clock mode
°81°-°8E° (Cont=0) (Cont=1)	 INTs 1-14 CK bits 1-14	 Processor interrupts CK bits, (event timer)
°8F° (Cont=0) (Cont=1)	 INT 15 INTREQ	 Signals active INT to c.p.u.
°90°-°94°	P0-P4 inputs	--
°95°	P5 input	Sys/Mon select
°96°	P6 input	--
°97°	P7 input	Height Zero set-up select
°98°-°9F°	P8-P15 inputs	--

Table 2 cont.

C.R.U. WRITE.

I/O bit	9901 bit def'n	Function
°80°	Control bit	
°81°-°8E° (Cont=0) (Cont=1)	 Mask bits 1-14 CK bits 1-14	 Interrupt mask bits CK bits, (load new value)
°8F° (Cont=0) (Cont=1)	 Mask 15 RST 2 ,	 Software generated reset
°90°	P0 output	Camera
°91°	P1 output	Ignition cut
°92°	P2 output	Parachute deploy
°93°	P3 output	Parachute jettison
°94°	P4 output	Airbag deploy
°95°-°97°	P5-P7 outputs	--
°98°	P8 output	(TLPORT) Soft. fail flag
°99°	P9 output	(TLPORT) Above set height
°9A°	P10 output	(TLPORT) Overflow flag
°9B°	P11 output	(TLPORT) S/V error
°9C°	P12 output	(TLPORT) A/D error
°9D°	P13 output	(TLPORT) Auto engaged
°9E°	P14 output	(TLPORT) Height lock
°9F°	P15 output	(TLPORT) Heading hold

Table 3

Regulated Power Supply Requirements.

This depends on the configuration of the system. The figures quoted are for an installation with 4K of EPROM, 256 bytes of RAM, front panel display disabled and all instruments installed.

Regulated voltage	Source volts	Average current	Regulator type	Rated capacity
+12v	+15.6V	0.35A	LM340T12	1.0A
+12V(gyro)	+15.6V	0.5A	LM340T12	1.0A
+6V	+15.6V	0.1A	LM217	1.5A
+5V	+8.4V	1.05A	LM323K	3.0A
-5V	-15.6V	0.1A	LM120H	0.2A
-6V	-15.6V	0.1A	LM237	1.5A
-12V	-15.6V	0.15A	uA7912	1.0A

In addition to this there is the servomotor supply, which is un-regulated, operating directly of 7.2V, 1.2AH battery stacks.

THESE ARE THE DE-COUPLED, SHORT PERIOD TRANSFER FUNCTIONS, USED TO REPRESENT THE AIRCRAFT DYNAMICS.

T.F.	22 m/s	40 m/s	50 m/s
ROLL RATE / AILERON DEFLECTION ($C_R(s)$)	$\frac{(-)45.61}{(s+10.64)}$	$\frac{(-)152.8}{(s+19.61)}$	$\frac{(-)237.6}{(s+24.39)}$
PITCH RATE / ELEVATOR DEFLECTION ($C_P(s)$)	$\frac{(-)21.21(s+2.276)}{s^2+4.764s+31.52}$	$\frac{(-)70.12(s+4.138)}{s^2+8.661s+104.21}$	$\frac{(-)109.56(s+5.173)}{s^2+10.827+162.84}$

A LINEARISED VERSION OF THE SERVO-ACTUATOR DYNAMICS IS GIVEN BY:

SERVO OUTPUT / SERVO DEMAND ($C_{SV}(s)$)	$\frac{20.0}{(S+20.0)}$	$\frac{20.0}{(s+20.0)}$	$\frac{20.0}{(s+20.0)}$

TABLE 4. THE AIRCRAFT TRANSFER FUNCTIONS, IN CONTINUOUS-TIME FORM.

THESE ARE THE RESULTS OF THE Z-TRANSFORMATION PROGRAMME, ZTRANS.FORTRAN.

T.F.	22 m/s	40 m/s	50 m/s
$G_{RSV}(z)$	$\dfrac{(-)0.00625(z+2.934)(z+0.2077)}{(z-1)(z-0.6125)(z-0.6065)}$		
$G_R(z)$	$\dfrac{(-)0.0131(z+0.9152)}{(z-1)(z-0.7664)}$	$\dfrac{(-)0.0408(z+0.8495)}{(z-1)(z-0.6125)}$	$\dfrac{(-)0.0612(z+0.8165)}{(z-1)(z-0.5435)}$
$G_{PSV}(z)$	$\dfrac{(-)0.00314(z+3.208)(z-0.917)(z+0.2295)}{(z-1)(z-0.6065)(z^2-1.747z+0.8053)}$		
$G_p(z)$	$\dfrac{(-)0.00649(z+0.9794)(z-0.9447)}{(z-1)(z^2-1.869z+0.8877)}$	$\dfrac{(-)0.0210(z+0.9628)(z-0.9017)}{(z-1)(z^2-1.747z+0.8053)}$	$\dfrac{(-)0.0325(z+0.9535)(z-0.8787)}{(z-1)(z^2-1.674z+0.7628)}$
$G_{SV}(z)$	"	$\dfrac{0.3935}{(z-0.6065)}$	"

TABLE 5. THE AIRCRAFT TRANSFER FUMCTIONS, IN DISCRETE TIME FORM. (SAMPLING RATE = 40 Hz)

T.F.	22m/s	40m/s	50m/s
$G_R(z)$ Error state	$\frac{(-)0.0002204(z+0.9879)}{(z-1)(z-0.9673)}$	$\frac{(-)0.0007312(z+0.9795)}{(z-1)(z-0.9406)}$	$\frac{(-)0.0011314(z+0.9746)}{(z-1)(z+0.9266)}$
Rate state	$\frac{(-)0.1402}{(z-0.9673)}$	$\frac{(-)0.4632}{(z-0.9406)}$	$\frac{(-)0.7149}{(z-0.9266)}$
Accel. state	$\frac{(-)45.61(z-1)}{(z-0.9673)}$	$\frac{(-)152.8(z-1)}{(z-0.9406)}$	$\frac{(-)237.6(z-1)}{(z-0.9266)}$
$G_{SV}(z)$ Servo state	"	$\frac{0.06059}{(z-0.9394)}$	"

TABLE 5a. THE AIRCRAFT TRANSFER FUNCTIONS, IN DISCRETE TIME FORM. (SAMPLING RATE = 320 Hz)

LABEL	PURPOSE OF COEFFICIENT	CALCULATION	RESULT (DECIMAL)	"MULT" COEFF	EXP
RDMSC	ROLL DEMAND SCALING	$\frac{(32752)}{(80)}\frac{(60)}{(127)}$	$0.75554 * 2^{8}$	°C16A°	8
RFGSC	ROLL FORWARD GAIN AND SCALING	$(0.4)\frac{(757.5)}{(10)}\frac{(80)}{(32752)}$	$0.0704011 * 2^{0}$	°12F2°	0
PDMSC	PITCH DEMAND SCALING	$\frac{(32752)}{(16.67)}\frac{(8)}{(127)}$	$0.96689 * 2^{7}$	°F785°	7
PFGSC	PITCH FORWARD GAIN AND SCALING	$(0.86)\frac{(757.5)}{(15)}\frac{(16.67)}{(32752)}$	$0.022105 * 2^{0}$	°05A8°	0
SSGN	FILT. STEADY STATE GAIN ADJUSTMENT	$\frac{(1.0-1.2759+0.33579)}{(1.0-1.7736+0.8187)}$	$0.6636 * 2^{1}$	°A9E1°	1
PCGB1	FILTER COEFF -(B1)	$-(-1.279)$	$0.63795 * 2^{1}$	°A350°	1
PCGB2	FILTER COEFF (B2)	(0.33579)	$0.33579 * 2^{0}$	°55F6°	0
PCGAB1	FILTER COEFF'S -(A1-B1)	$-(-1.7736+1.2759)$	$0.4977 * 2^{0}$	°7F69°	0
PCGAB2	FILTER COEFF'S (A2-B2)	(0.8187-0.33579)	$0.48291 * 2^{0}$	°7BA0°	0
HDMSC	HEIGHT DEMAND SCALING	$\frac{(32752)}{(255)}$	$0.50172 * 2^{8}$	°8070°	8
HFGSC	HEIGHT FORWARD GAIN AND SCALING	$\frac{(1000)}{(32752)}\frac{(127)}{(20)}$	$0.19388 * 2^{0}$	°31A2°	0

TABLE 6 CALCULATION OF THE CONTROL ALGORITHM COEFFICIENTS.

REFERENCES.

REFERENCES.

1 R.P.V's: Robot aircraft today. Janes book 13.
J. Taylor,K. Munson Macdonald and Janes, 1977

2 Development of a mini R.P.V.
R. Stephenson Aerospace, BAe Research Journal, 1980

3 RAE PCM Telecommand system.
M. Hawkins RAE report, 1979

4 The RAE lightweight PCM telemetry system for R.P.V's.
M. Hawkins RAE report, 1979

5 The TMS 9900 microprocessor data manual.
Texas Instruments incorporated, 1976

6 The TMS 9901 programmable system interface manual.
Texas Instruments incorporated, 1977

7 The TM 990-100M microcomputer users guide.
Texas Instruments incorporated, 1978

8 CMOS interfacing simplified,
Cosmos/Mos Integrated Circuits
D. Blandford, A. Bishop RCA corporation, 1978

9 The ADC80AGZ datasheet.
Burr-Brown Research corporation, 1978

10 The Voltage Regulator handbook.
National Semiconductor corporation, 1975

11 PDP/11 screen editor manual.
Digital Equipment Corporation, 1978

12 Tibug monitor listing TM/990/401-1
Texas Instruments incorporated, 1978

13 Stabileye control system design.
J.E.Bailey BAe report ST.19758, 1978

14 A control system design study for Stabileye Mk3.
W.W.Black BAe report BT. 11353, 1981

15 Aircraft dynamic stability and response.
A.W.Babister Pergamon press, 1980

16 Estimated derivatives for Stabileye Mk3.
A.E.R.I.Rogers BAe report, 1980

17 Digital control systems.
B.C.Kuo Holt, Rinehart and Winston, 1980

18 Discrete time and computer control systems.
Cadzow and Martens Prentice Hall 1970

19 Theory and application of the z-transform method.
E.I.Jury Robert E. Krieger 1964

20 Mathematical modelling and simulation
for engineers and scientists.
J.M.Smith John Wiley 1977

21 Servo torque load data from Mk3 wind tunnel trials.
D.A.Carruthers BAe report 821/RPV/6619/23, 1981

22 Skyleader SRC44b (rotary servo) specification.
Skyleader radio control Ltd. 1981

23 Digital control using microprocessors.
P.Katz Prentice Hall International, 1981

24 Digital algorithms for prediction,
differentiation and integration.
W.Forsythe Trans Inst. MC voll no.1, 1979

25 Introduction to digital filters.
J.Terrel Macmillan press, 1980

26 Report on the flight of Stabileye no. 43,
(fitted with the DFCS).
S.N.Gittens BAe report 821/RPV/7078, 1983

27 Use of non-linear correcting devices of switch-type to improve the quality of second order automatic control systems.
S.V.Emelyanov Automat i Telemekh 1959, 20, (7) (In Russian).

28 Design principles of V.S.S. for control of non-stationary plants.
S.V.Emelyanov et. al. Control of complex systems 1966 (session 40)

29 Method of stabilising automatic control systems with a varying structure without using errors derived from the signal.
S.V.Emelyanov, V.A.Taran Engineering Cybernetics 1963 no. 1

30 Systems having a variable structure which use continuous differences in the control law.
A.V.Bakakin, V.A.Taran Engineering Cybernetics 1966 no. 2

31 A design principle for stabilisation systems.
E.A.Barbashin, E.I.Gerashenko Automat i Telemekh 1965, 26, (6)

32 Compensation for the forced component of motion in variable structure systems.
V.I.Utkin Engineering Cybernetics 1965 no. 4

33 Quasi- invariant control of forced motion in variable structure systems.
V.I.Utkin Engineering Cybernetics 1966 no. 5

34 Realisation and application of a new non-linear attitude control system.
W.Sobotta Proc. 4th IFAC Symp. Control in space 1974

35 Variable structure systems with sliding modes.
V.I.Utkin Survey paper, IEEE Trans. on Auto. Control 1977

36 Control systems of variable structure .
U.Itkis John Wiley, 1976

37 Sliding modes and their application in variable structure sytems.
V.I.Utkin MIR,Moscow 1978

38 Adaptive and optimal variable structure control systems.
S.P.Maslen Ph.D. Thesis University of Bath 1981

39 Reduced order switching functions in variable structure control systems.
B.A.White IEE Proc. D. Vol 130, no. 2 1983

40 Reachability in variable structure control systems.
B.A.White ,P.M.Silson IEE Proc. D. Vol 131, no.3 1984

41 Adaptive control application to aircraft.
E.G.Ryanski *

42 Perspectives on the application of adaptive control to aircraft systems.
G.Kreisselmeier *

* Both papers included in : Applications of adaptive control.
Editted by: K Nahrendra, R Monopoli Academic press 1980

43 A new model reference adaptive scheme for aircraft flight control systems.
F.R.Gill, D.Maclean Trans. Inst. M.C. Voll, no.1 1979

44 The case for adaptive autostabilisation of aircraft.
A.R.M.Norton Control May 1963

45 Digital control of high performance aircraft using adaptive estimation techniques.
H.F.Van Landingham, R.L.Moose IEEE Trans. Aero. and Elec. Vol. AES-13, no.2 1977

46 Control with a multiplicative mode.
R.R.Mohler, R.E.Rink Journal of Basic engineering June 1969

47 Direct Digital control of thyristor drives.
R.T.Lipczynski Ph.D. Thesis University of Bath 1977

48 Dynamic switching of Type I, Type II structures in tracking servosystems.
U.Itkis IEEE Trans. Auto. control Vol AC28, no.4 1983

49 Analysis and synthesis of high gain and variable structure feedback systems.
K.Young Ph.D. Thesis University of Illinois 1978

50 The response of variable structure systems to reference input demands.
B.A.White, D.K.K.Wong Unpublished paper,University of Bath 1983

51 Application of variable structure systems to the stabilisation of variable parameter controlled objects with restrictions imposed on the motion of the controlling member.
A.V.Bakakin, M.A.Bermant, V.B.Ezerov
Automat i Telemekh 1964, 25, (7)

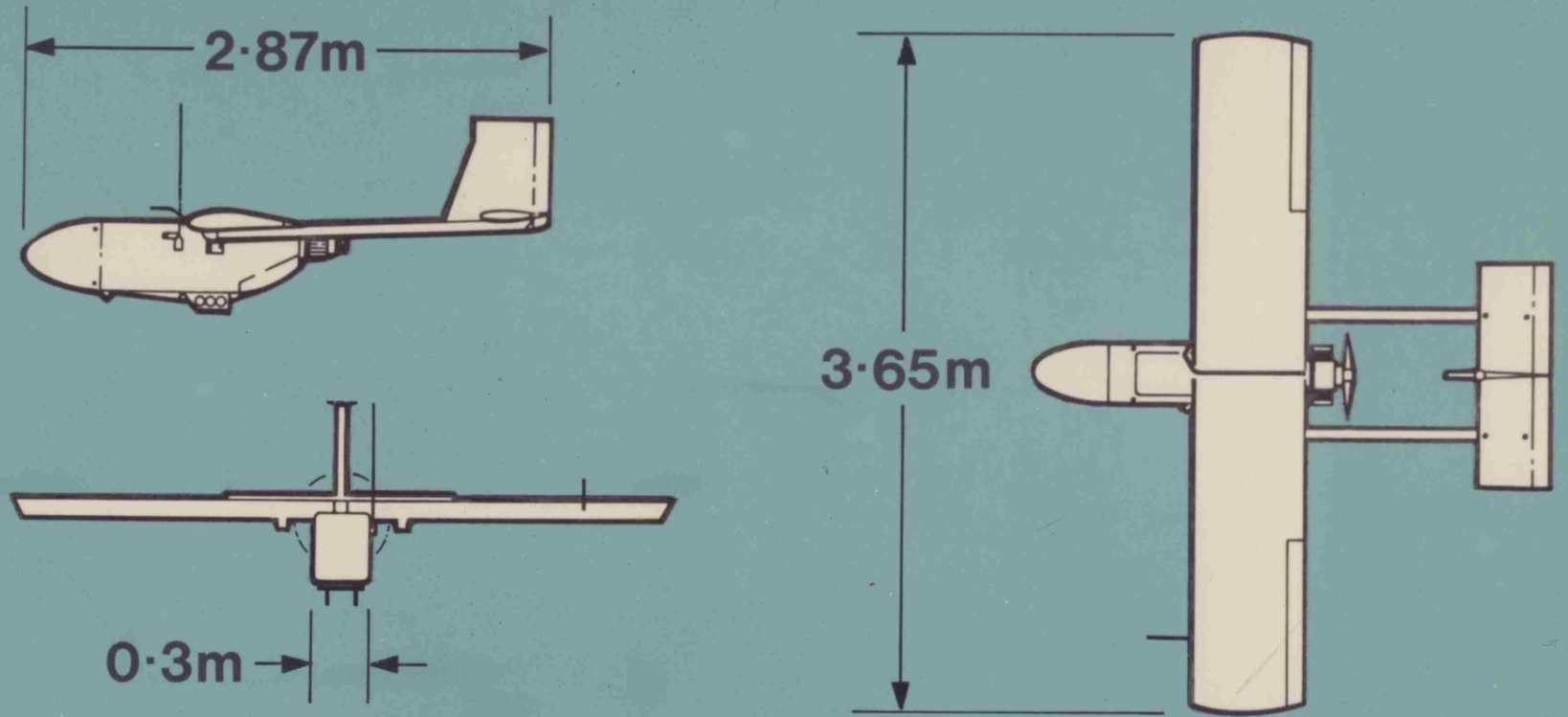
52 Himat on-board flight computer system architecture and qualification.
A.F.Myers, M.R.Earls AIAA Computers in Aerospace conf. paper 81-2107 San Diego 1981

53 Development, flight test and application of the R.P.V. control law concepts for microprocessor based computers.
M.Wooley,Teledyne Ryan. Proc. Int. conf. on R.P.V's 1979

54 Design of a DFCS for Stabileye: Volume 2.
S.N.Gittens, British Aerospace Tech. report no. 821/RPV/7173

FIGURES.

General Arrangement



Launch weight = 66 kg

Datum payload = 20 kg

B4536

Fig. 1 General view of Stabileye mkIII

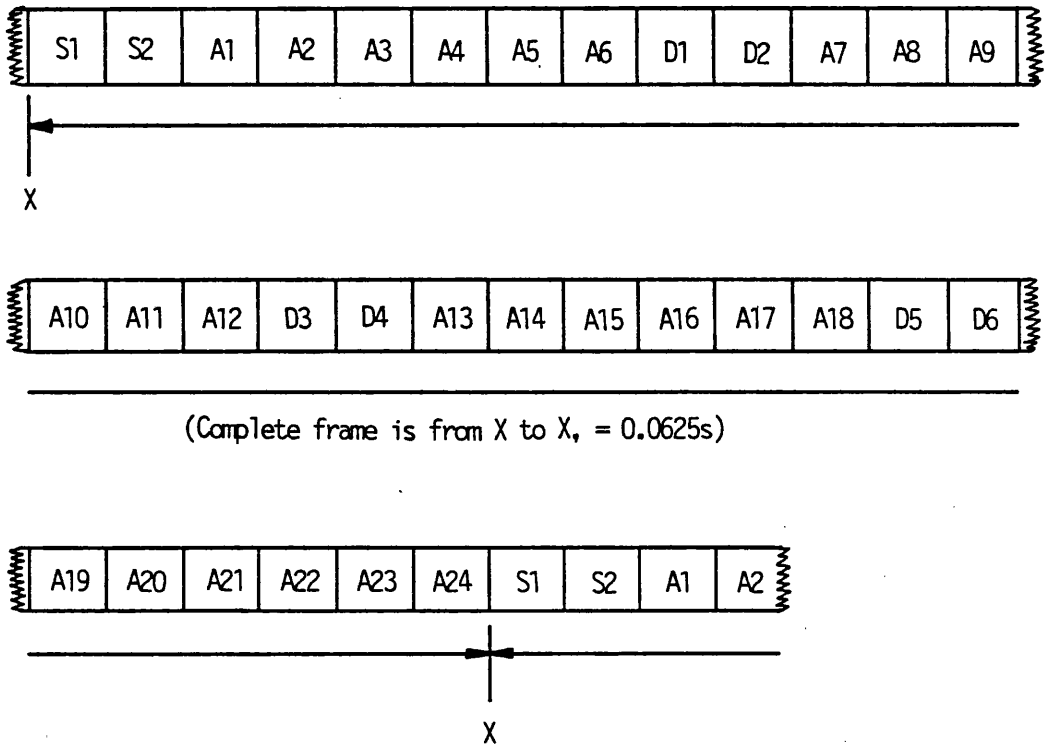
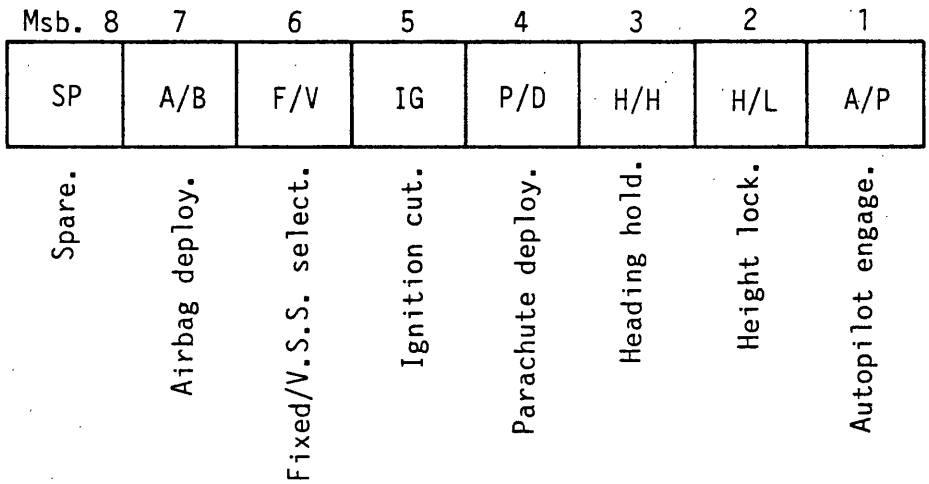
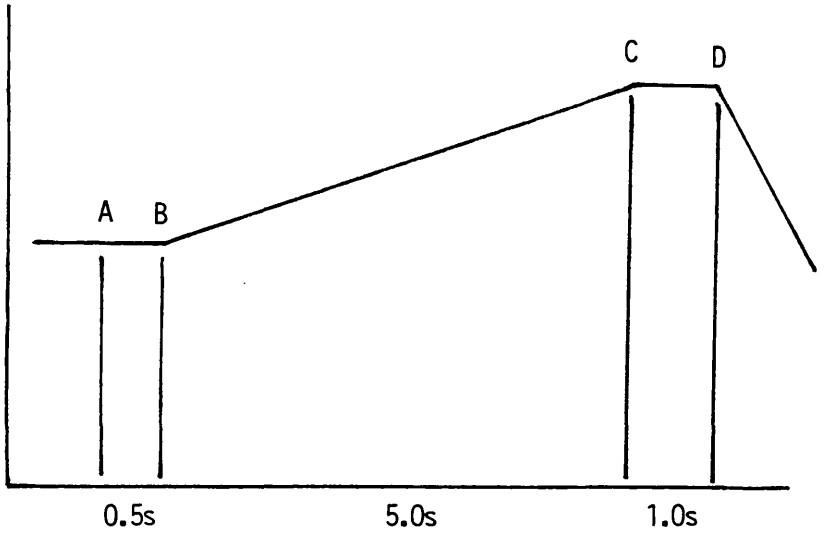


Fig. 2 The 32-byte Telemetry frame.



Note: all functions are active-true.

Fig. 3 The Telecommand status word.



Activities:

A - Receiver loses telecommand synch.

B - After loss of synch. for period greater than 0.5s, the software failsafe routine is started:
Height lock engaged, full height demand (1000m).
Full throttle demanded.
Roll demand set to 10 degrees.
Gives spiral climb to 100m, (this phase is omitted if the aircraft is above 100m at start). Then :
Roll demand set to 0 degrees.
5 second count started.

C - After 5s delay:
Pitch demand set to 0 degrees.
Ignition cut.
Throttle closed.

D - After further 1s delay :
Parachute deployed.
Airbag deployed.

Note : Spiral climb can be omitted if not required.

Fig. 4 The Software Failsafe procedure.

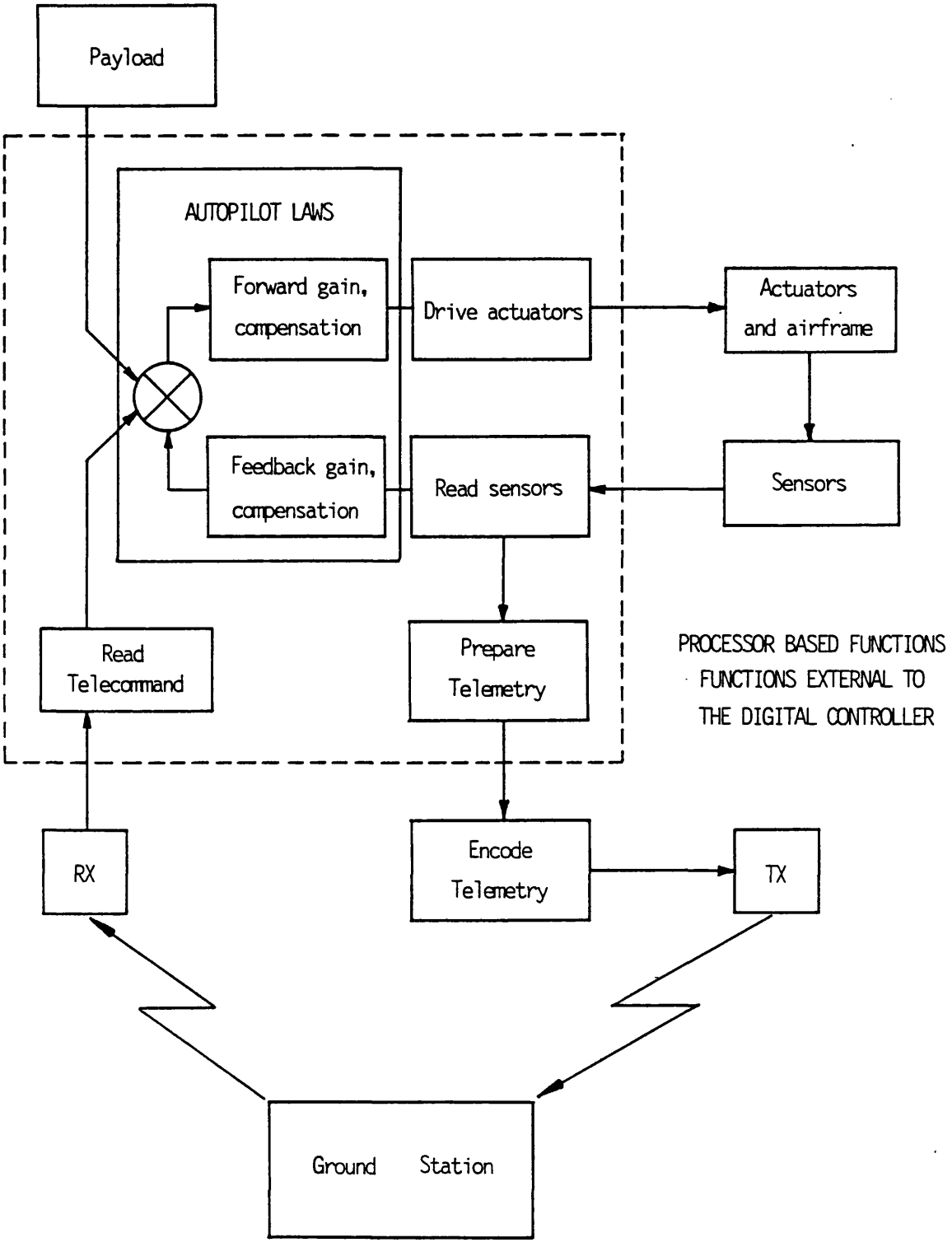


Fig. 5 Functional block diagram of the flight control system.

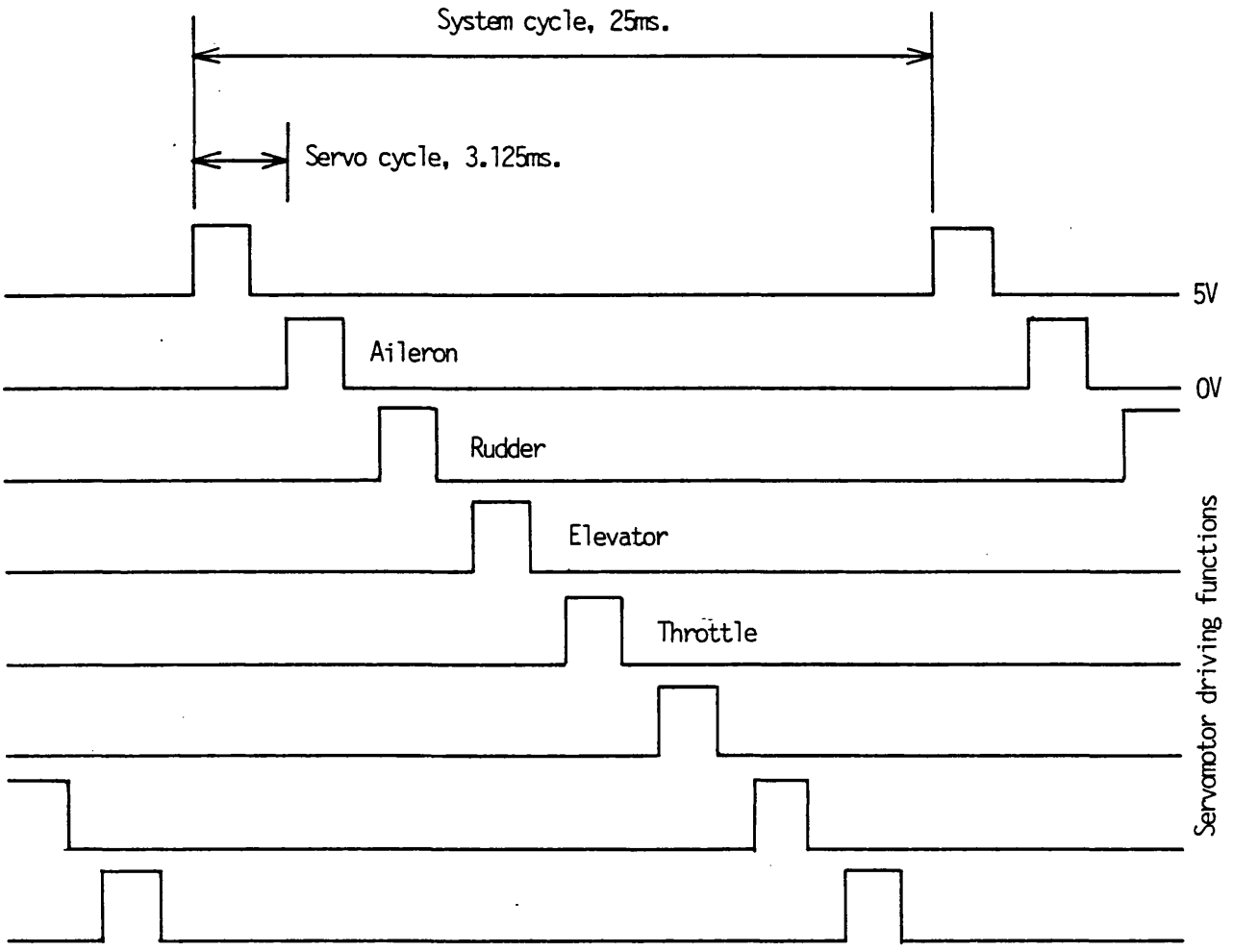


Fig. 6 Interleaving the Servomotor channels

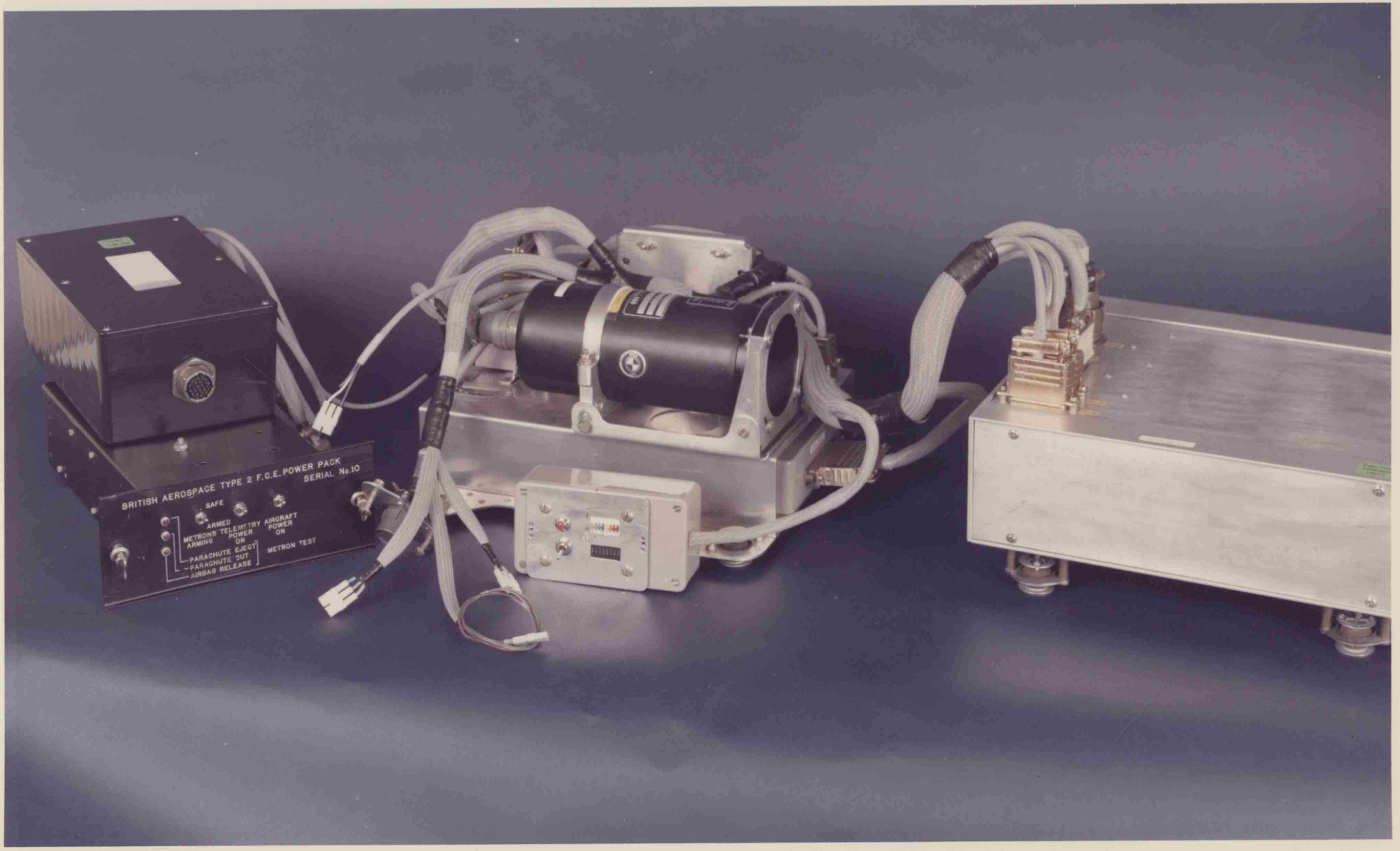


Fig. 7 The Digital Flight Control system

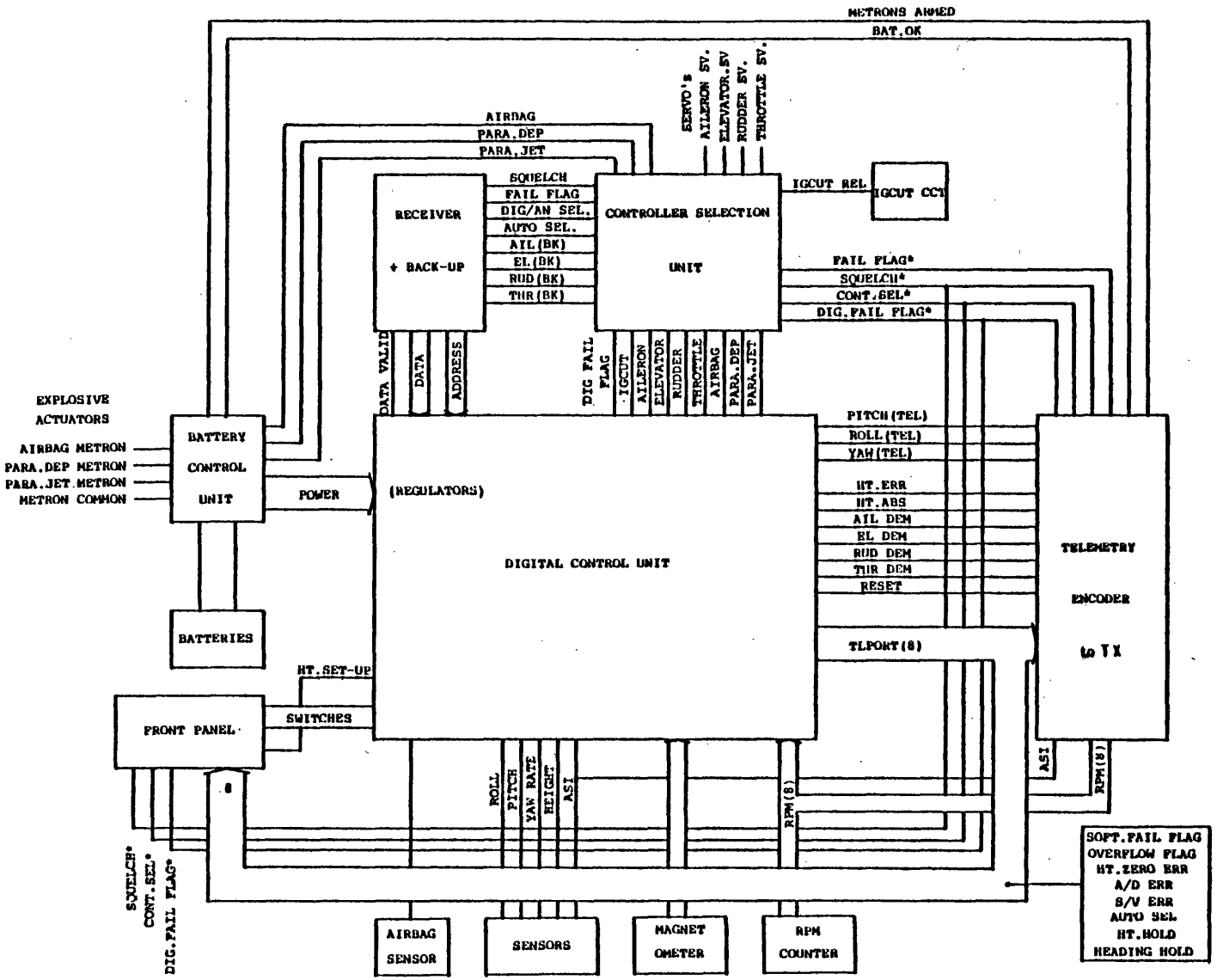


Fig. 8 Interconnection of the Digital flight control electronics

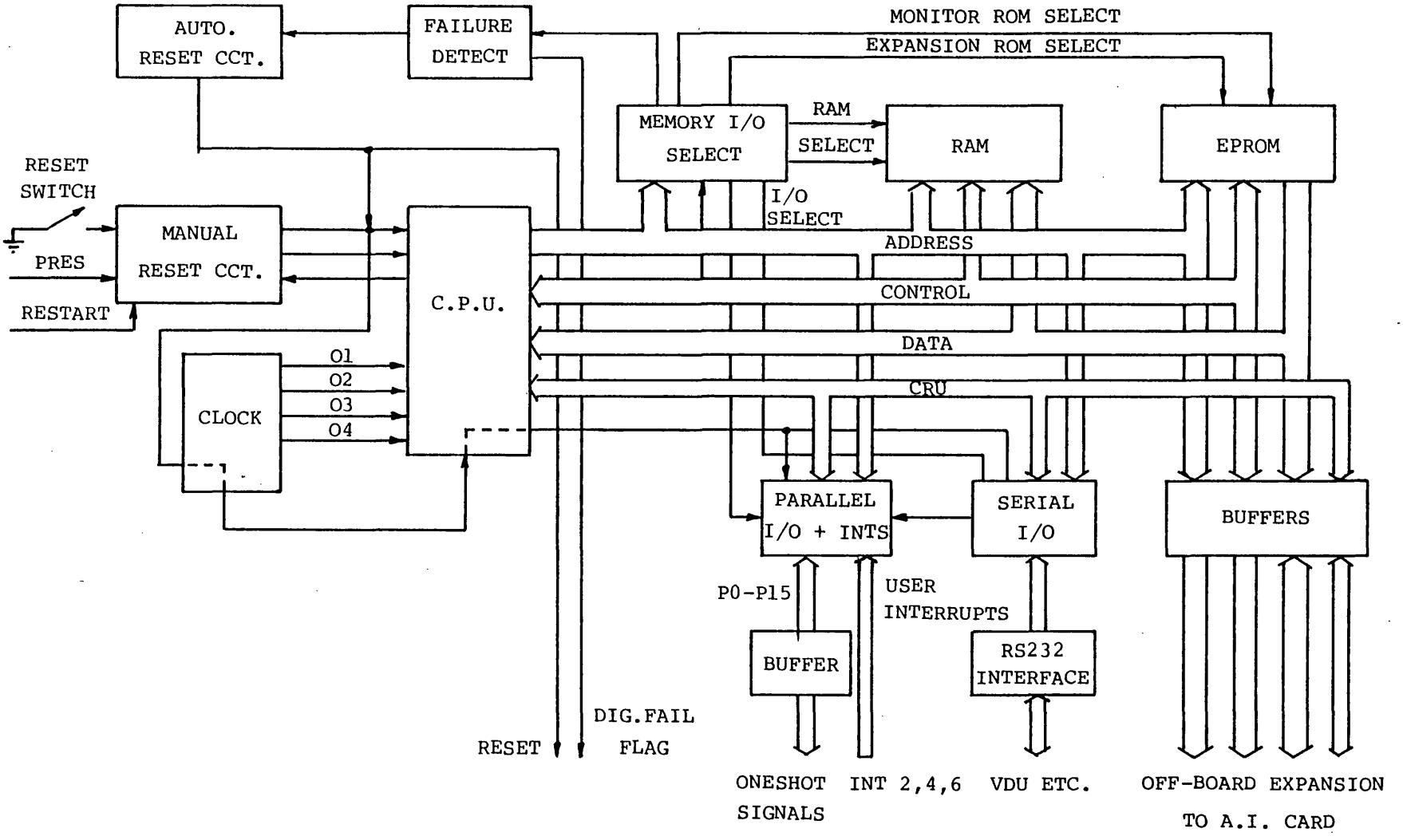


Fig. 9 Block diagram of the c.p.u. card

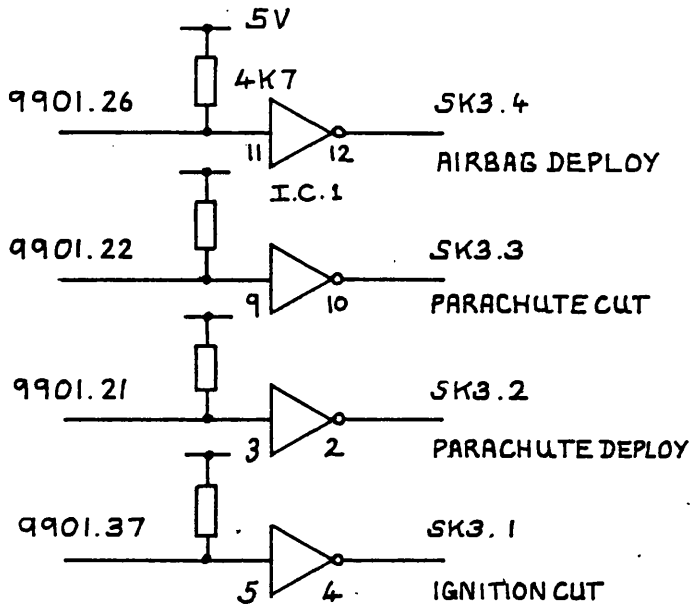


FIG. 10A

THE I.O. BUFFER CCT.

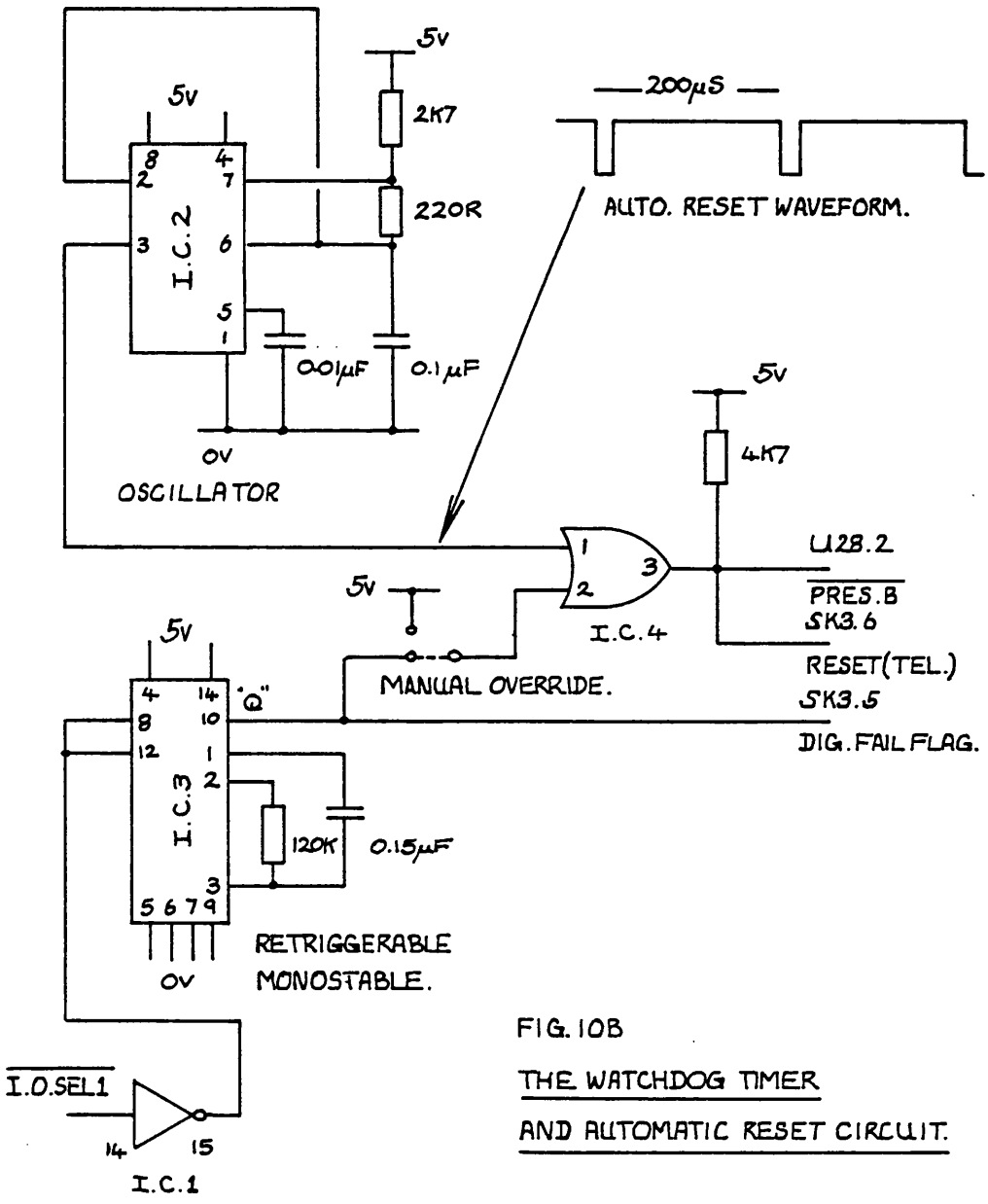


FIG. 10B

THE WATCHDOG TIMER

AND AUTOMATIC RESET CIRCUIT.

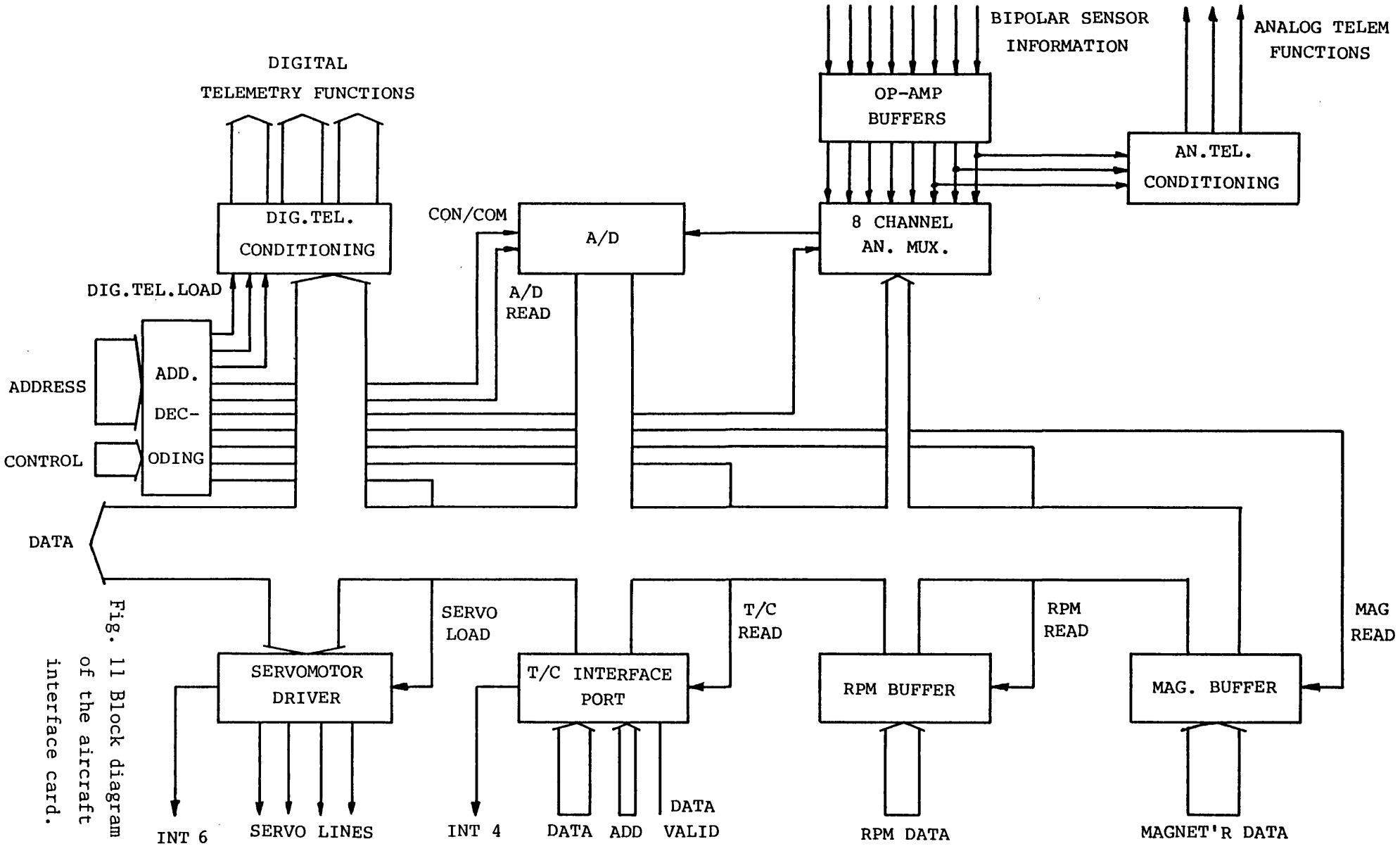
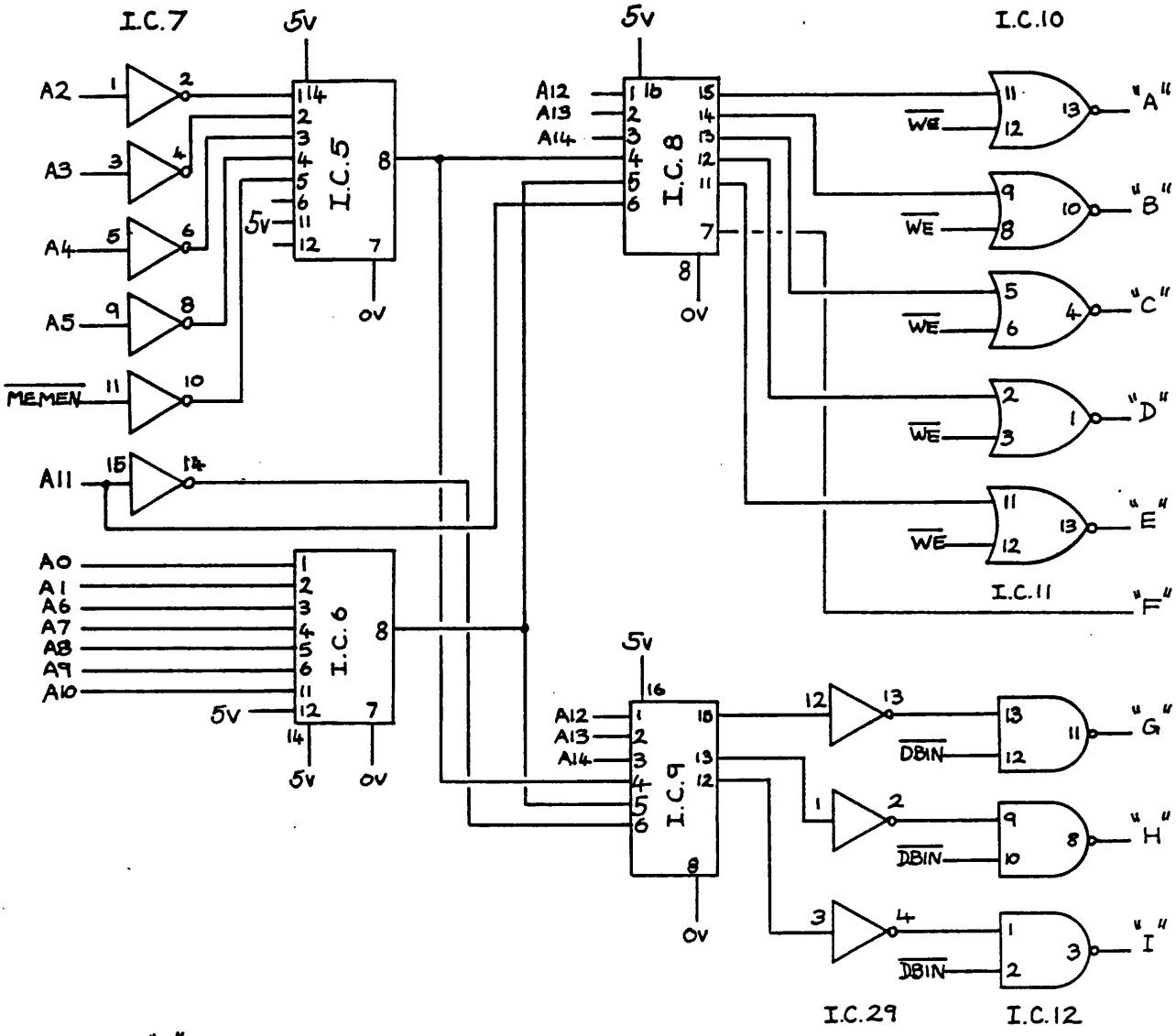


Fig. 11 Block diagram of the aircraft interface card.



"A"	C3F0	SERVO. DRIVER
"B"	C3F2	TLPORT + AILDEM
"C"	C3F4	A.D. ADDRESS LOAD
"D"	C3F6	CONVERT COMMAND
"E"	C3F8	SPARE
"F"	C3FE	RUDDEM + ELDEM
"G"	C3E0	T.C. PORT READ
"H"	C3E4	A.D. READ
"I"	C3E6	SPARE

FIG. 12 ADDRESS DECODING CCT.

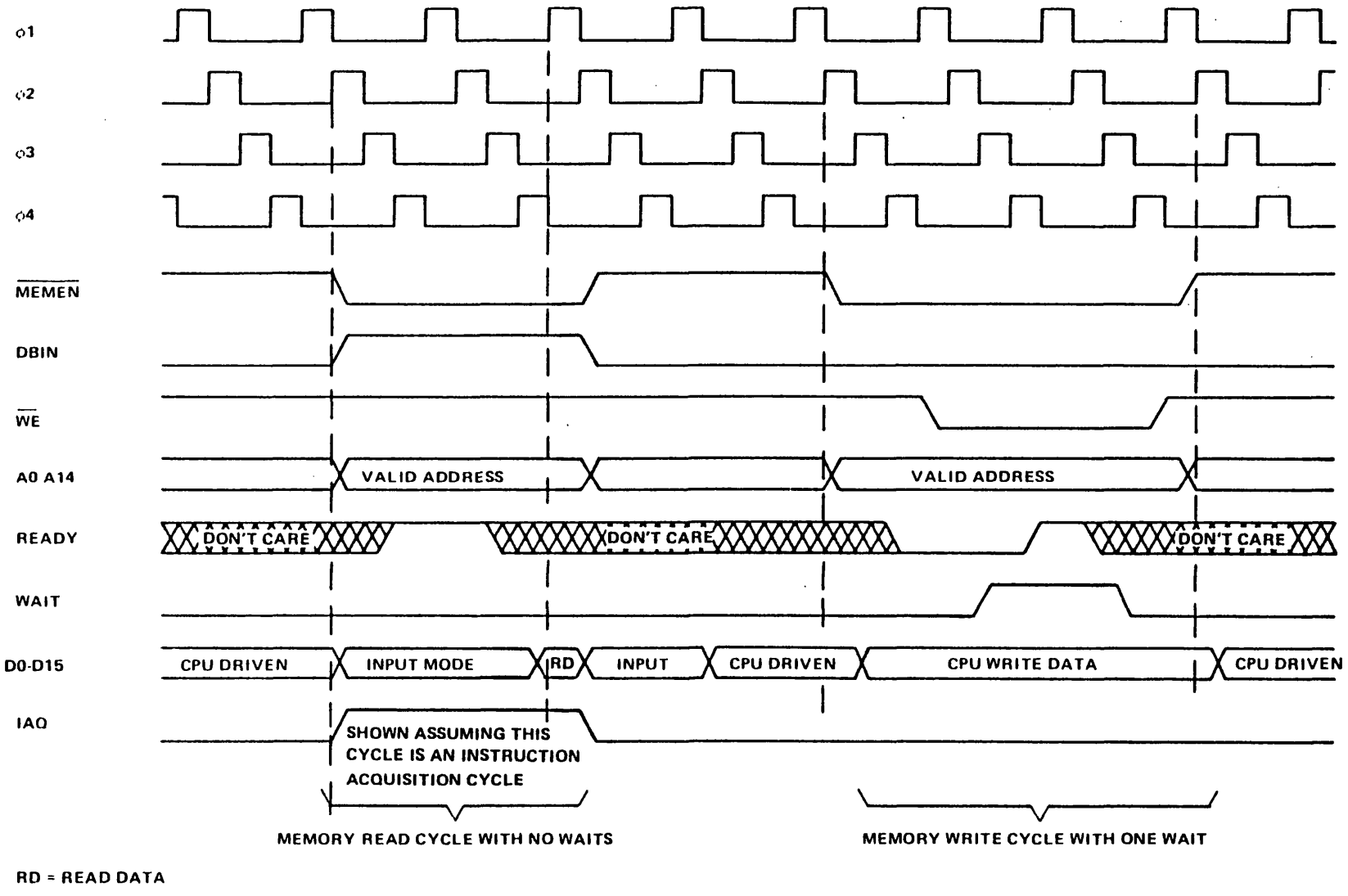


Fig. 13 TMS 9900 memory bus timing.

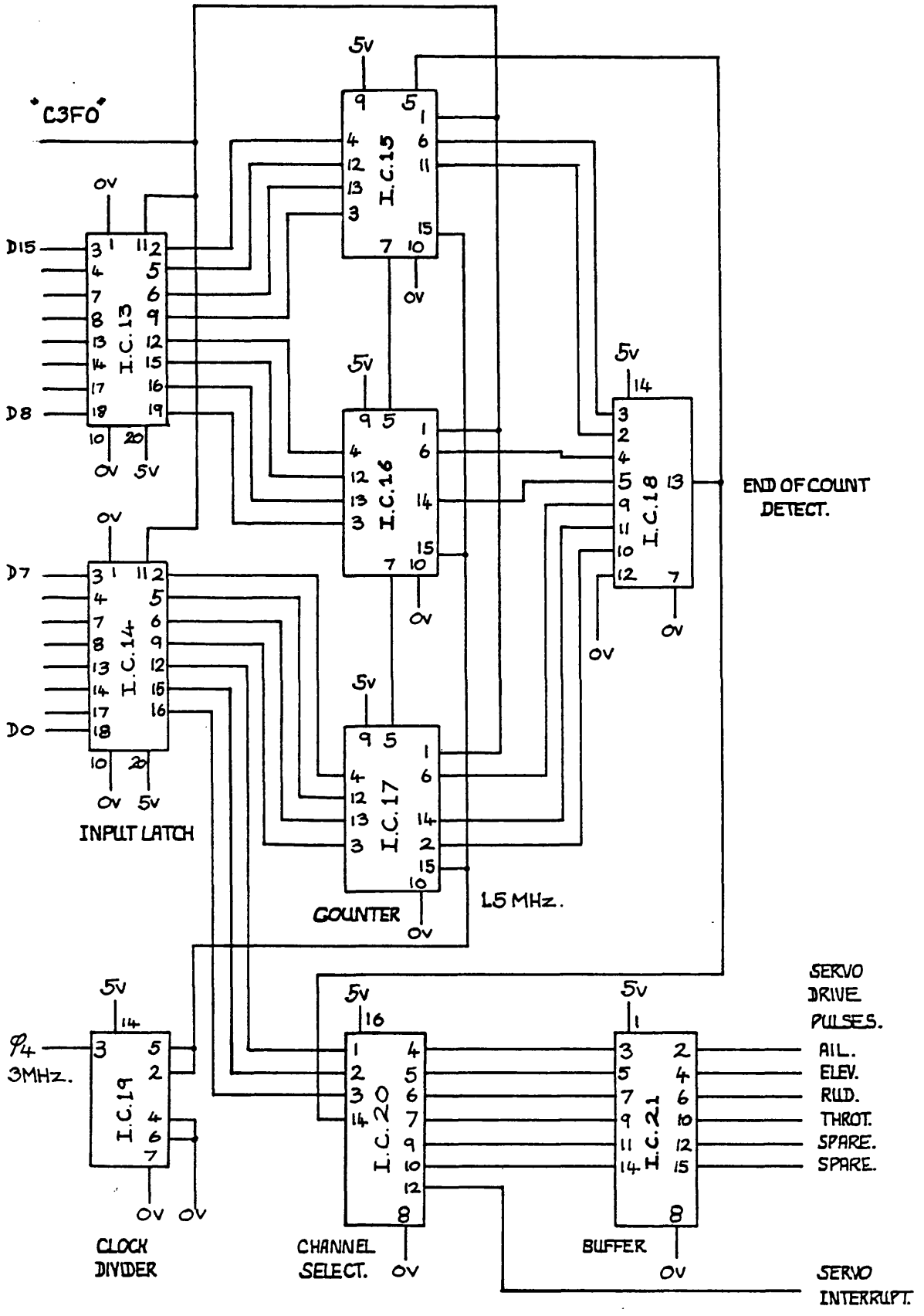
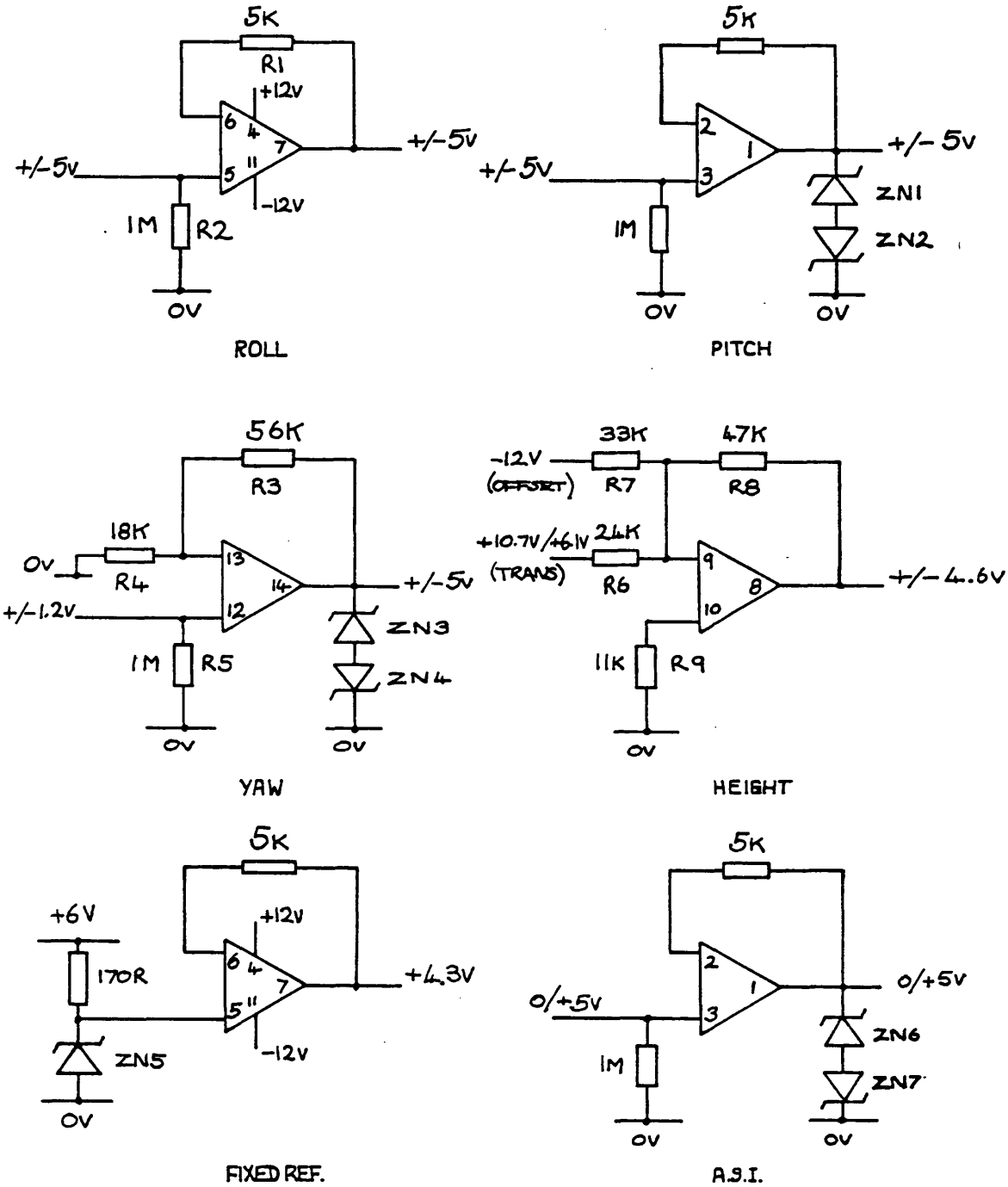


FIG. 11. EIGHT CHANNEL SERVOMOTOR DRIVER CCT.

OP.AMPS. ARE I.C. 22 I.C. 23



ZN 1.2.3.4.6.7. ARE 4V7.

ZN 5. IS 4V3.

FIG.15 SENSOR INPUT BUFFER CIRCUITS.

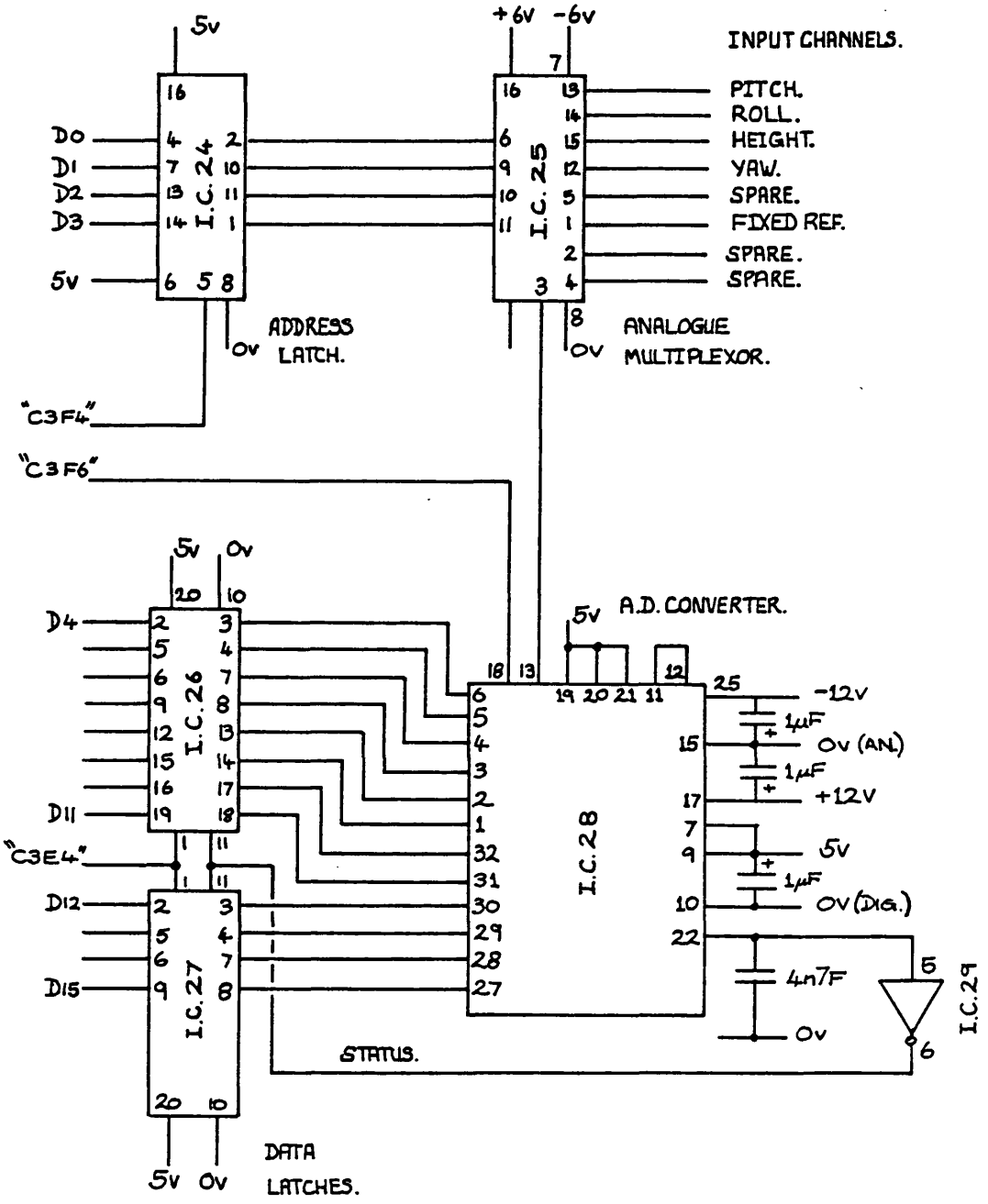


FIG. 16 ANALOGUE MULTIPLEXOR AND A.D. CONVERTER CCT.

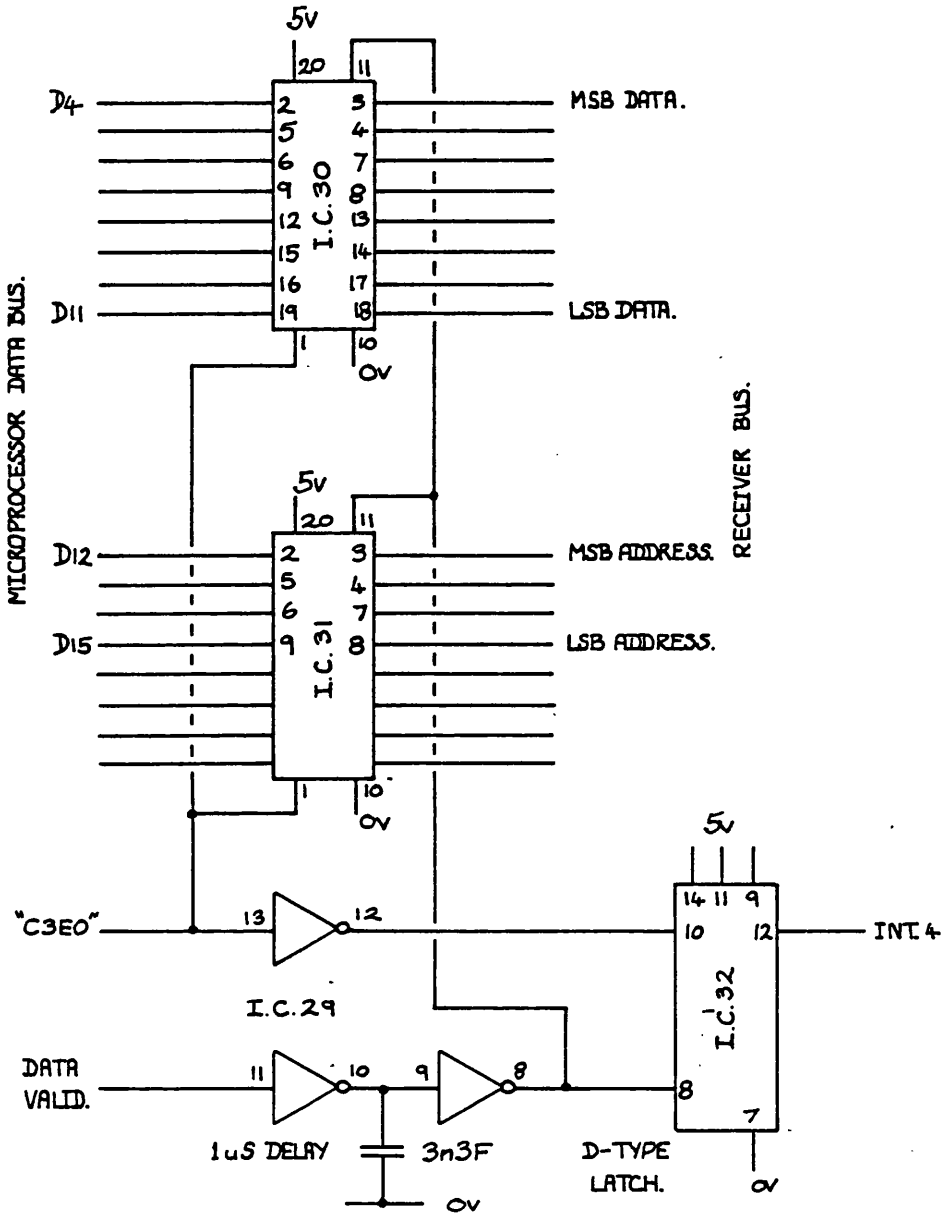


FIG.17 THE TELECOMMAND INTERFACE PORT.

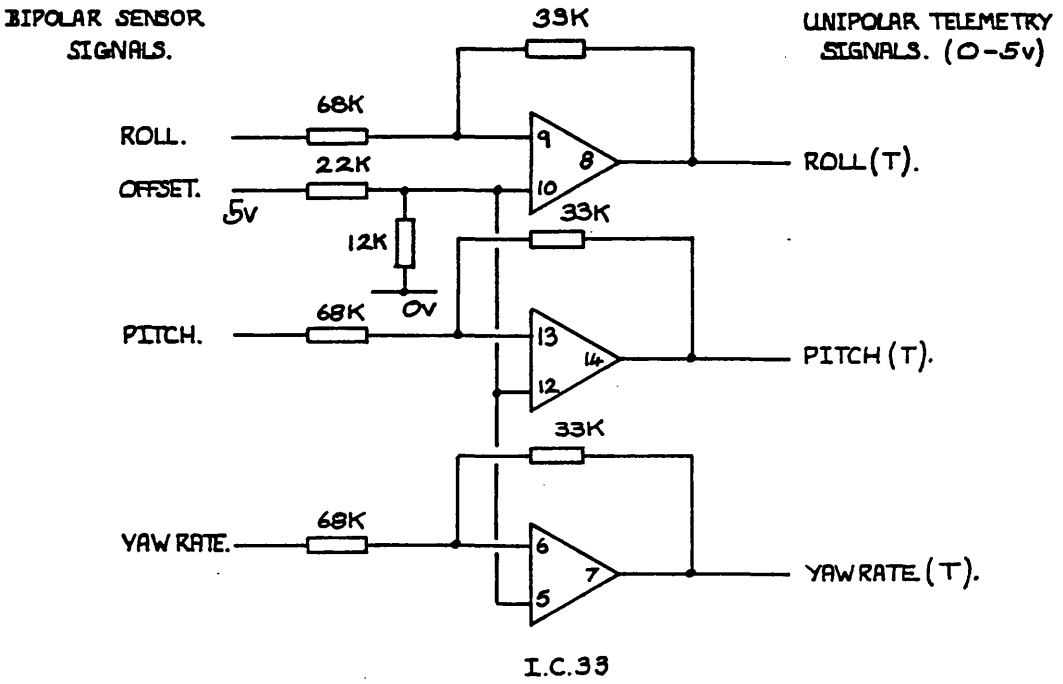


FIG. 18 ANALOGUE TELEMETRY SIGNAL CONDITIONING.

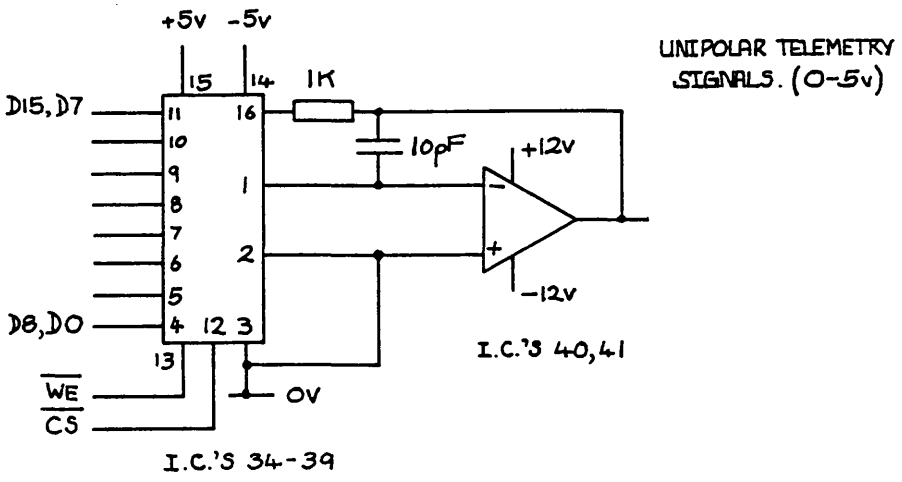


FIG. 19 D.A. TELEMETRY CONVERSION CCT.

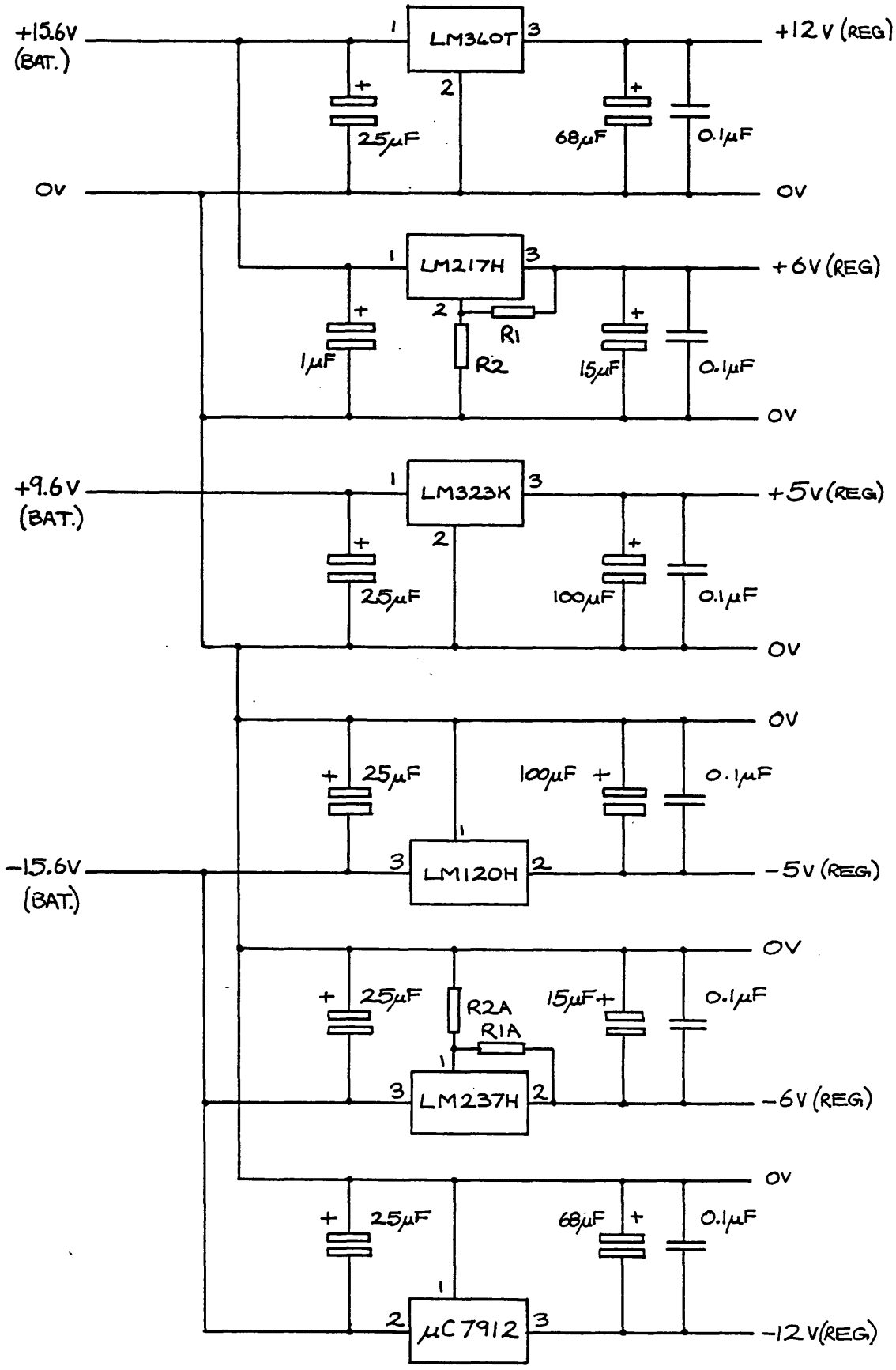


FIG. 20 THE VOLTAGE REGULATOR CCTS.

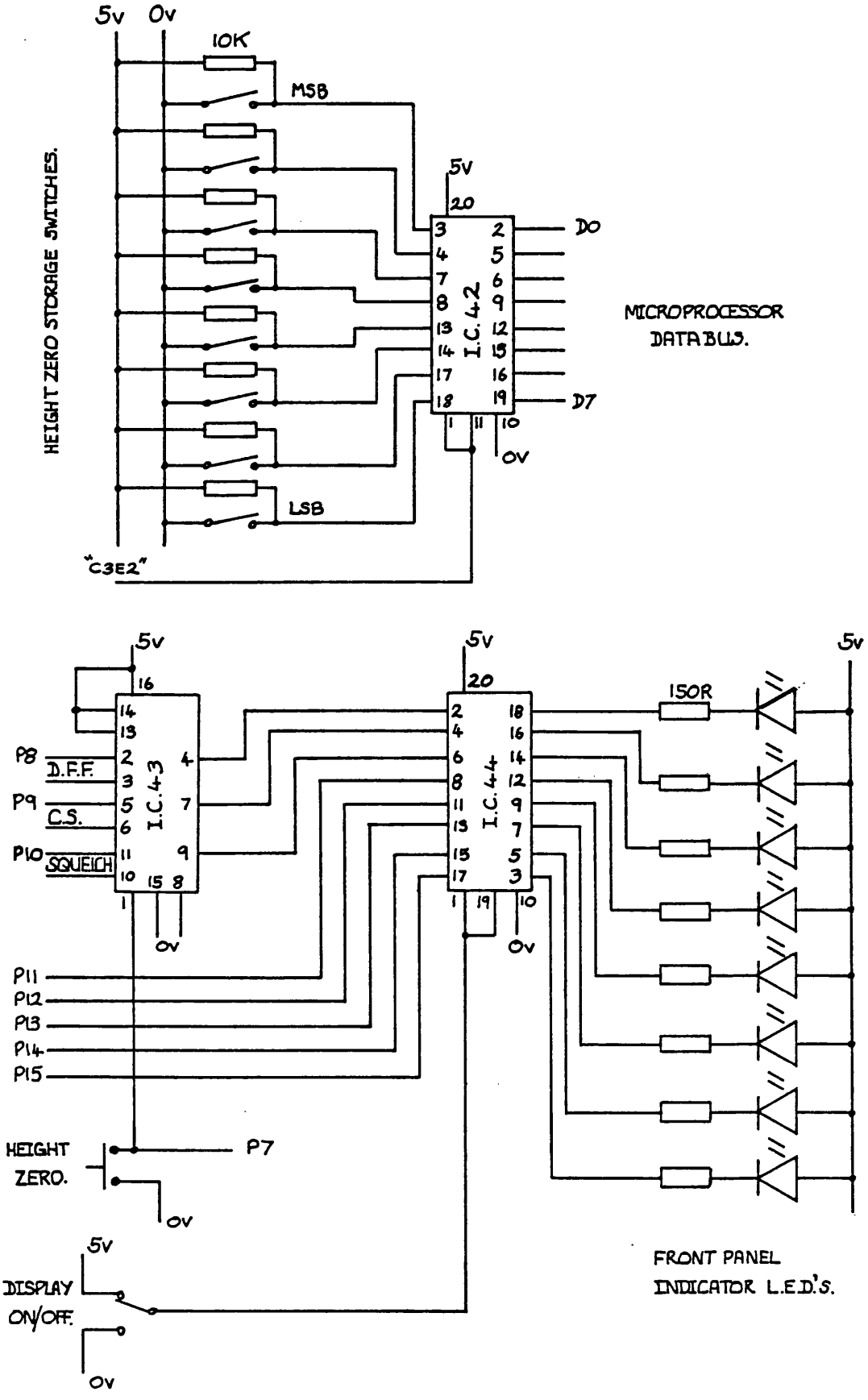


FIG. 21 THE HEIGHT ZERO STORAGE.

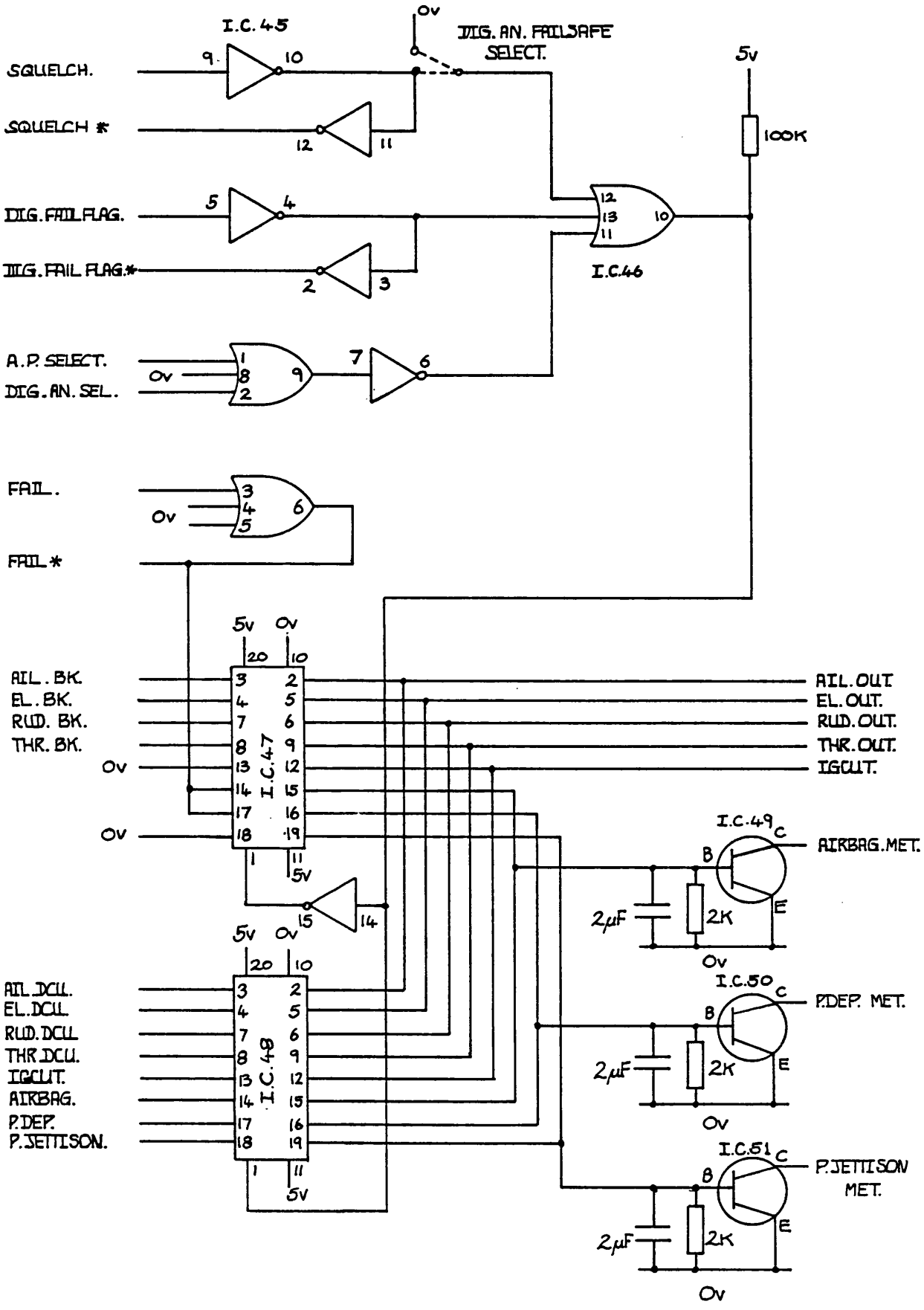
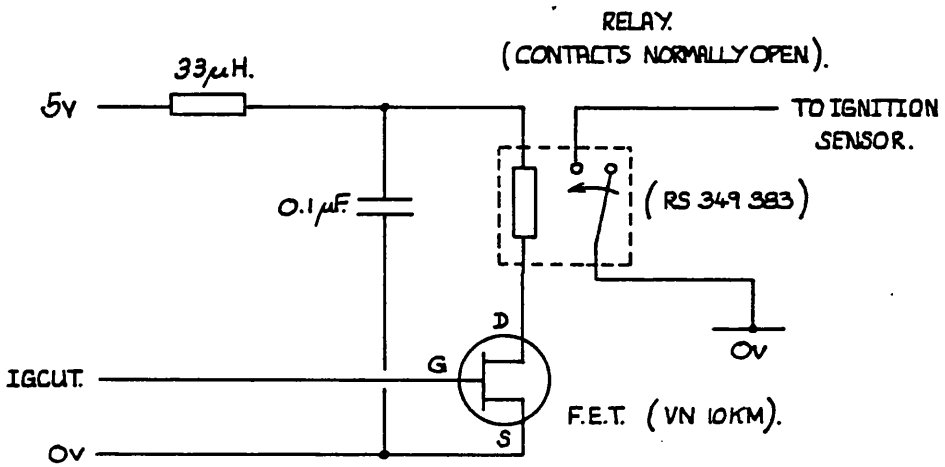
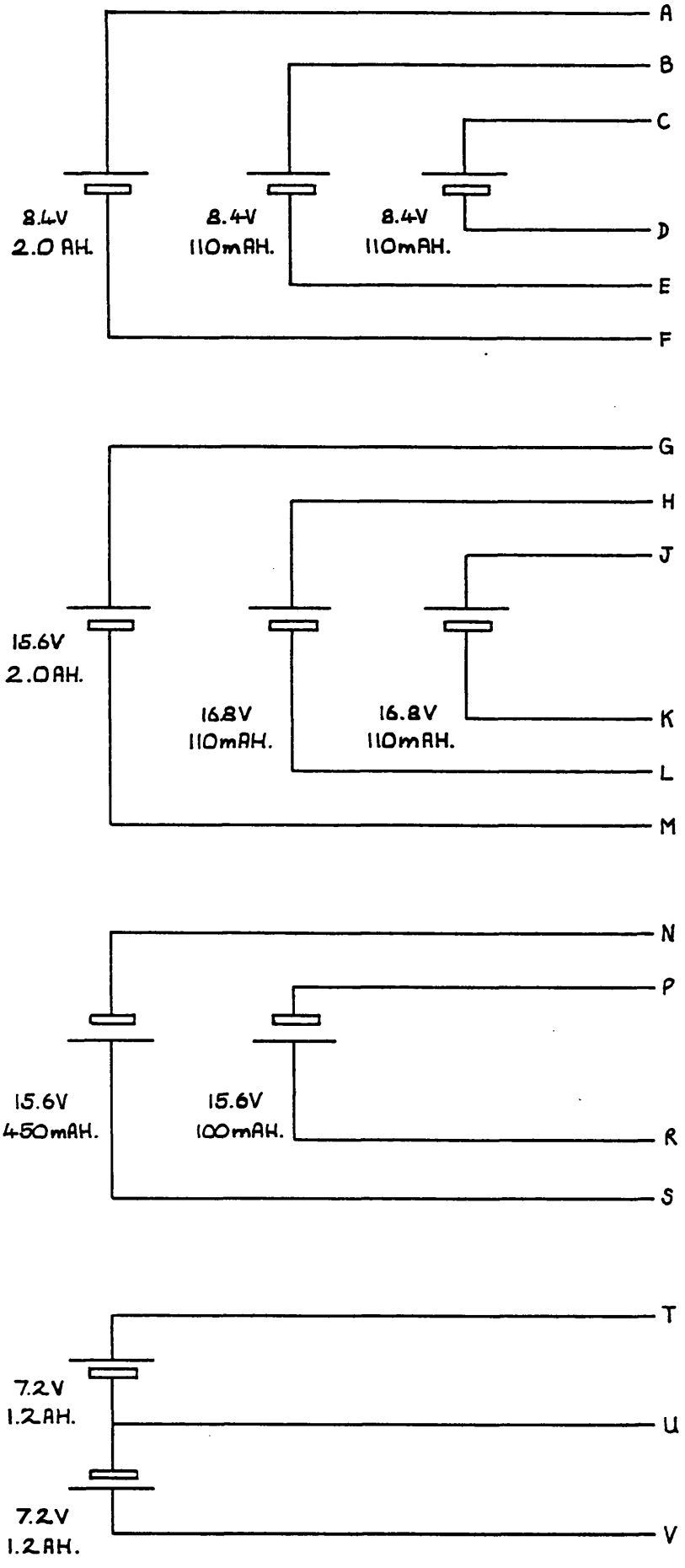


FIG. 22. THE CONTROLLER SELECTION UNIT.



IGNITION IS INHIBITED WHEN RELAY CONTACTS ARE CLOSED.

FIG. 23. THE IGNITION CUT ACTUATOR.



LETTERS REFER TO CONNECTIONS AT BATTERY SUPPLY CONTROL UNIT. (SEE FIG. 25.)

FIG. 24 THE BATTERY STACKS.

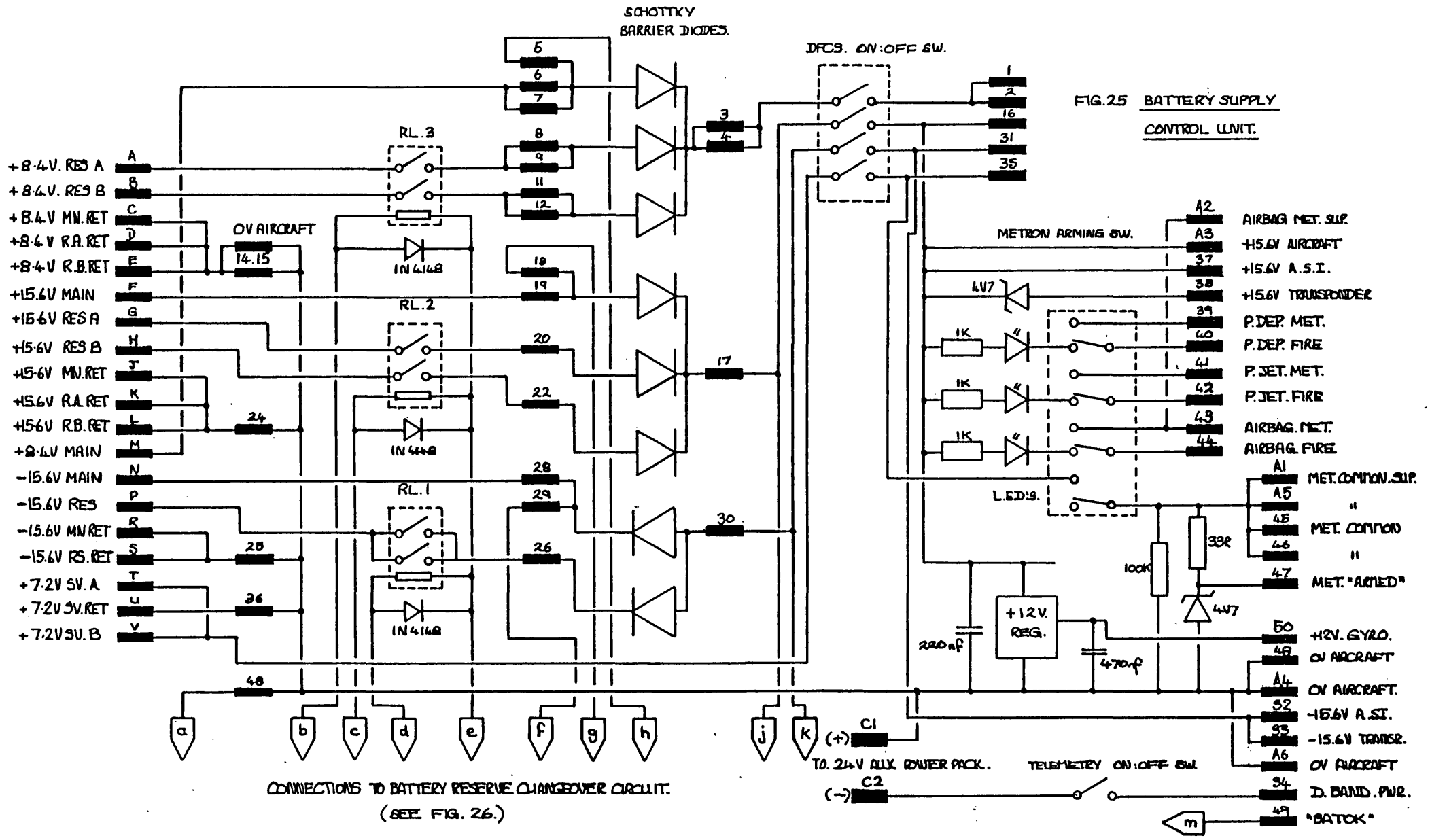


FIG.25 BATTERY SUPPLY CONTROL UNIT.

CONNECTIONS TO BATTERY RESERVE CHANGEOVER CIRCUIT.
(SEE FIG. 26.)

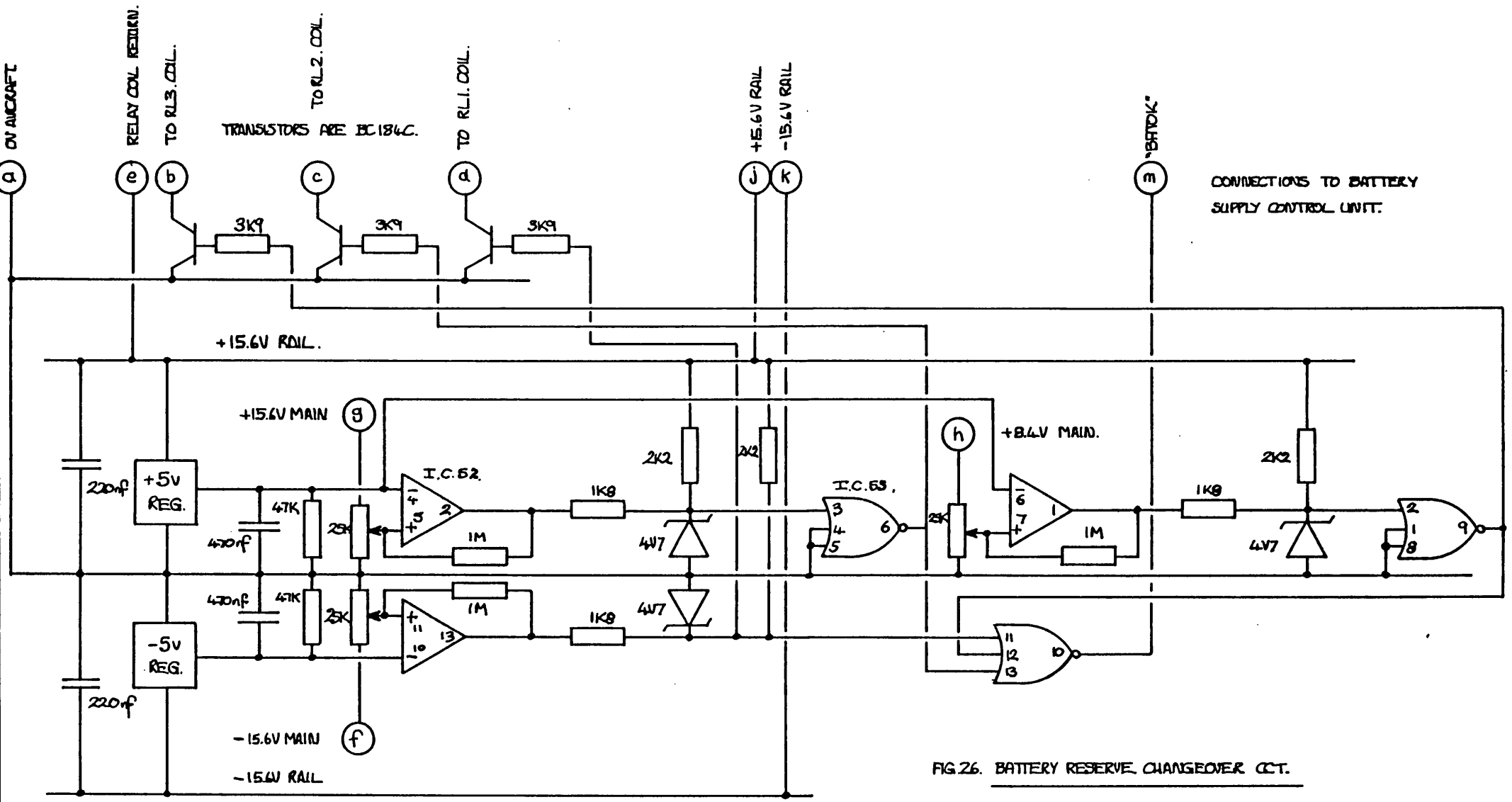


FIG.26. BATTERY RESERVE CHANGEOVER CCT.

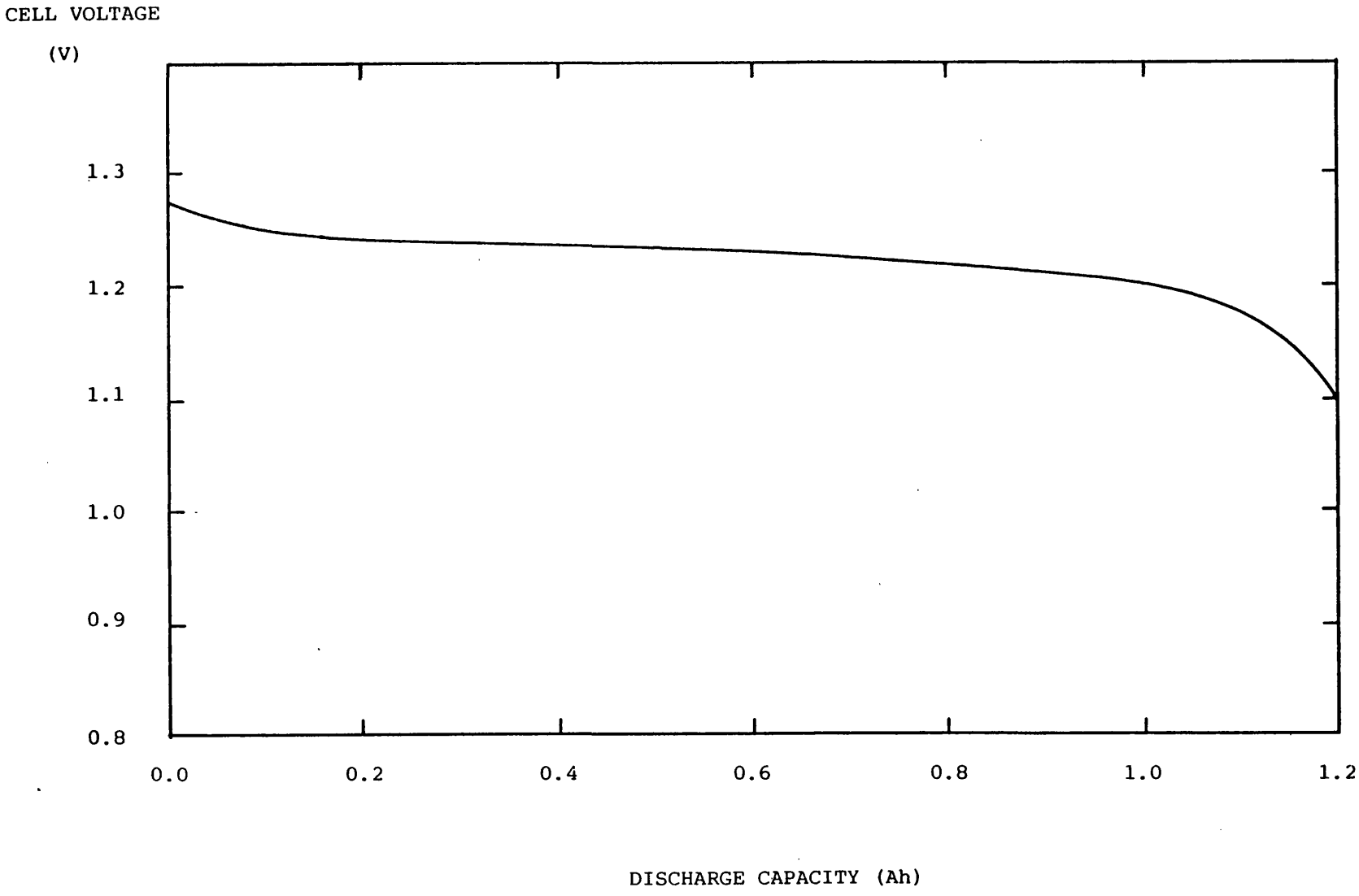


FIG. 27

Nickel-Cadmium cell discharge characteristic. (For a 0.24 Ah cell).

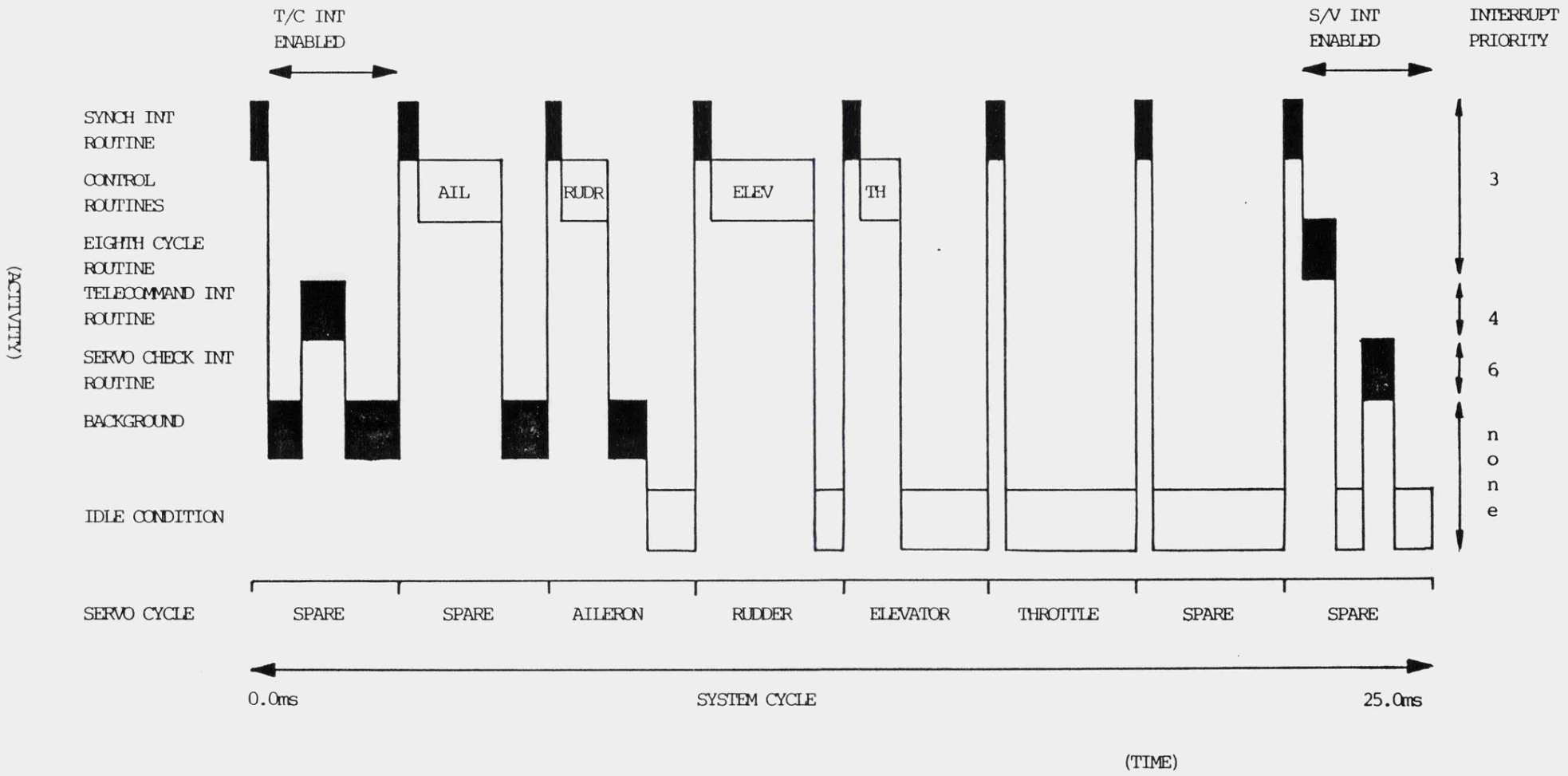
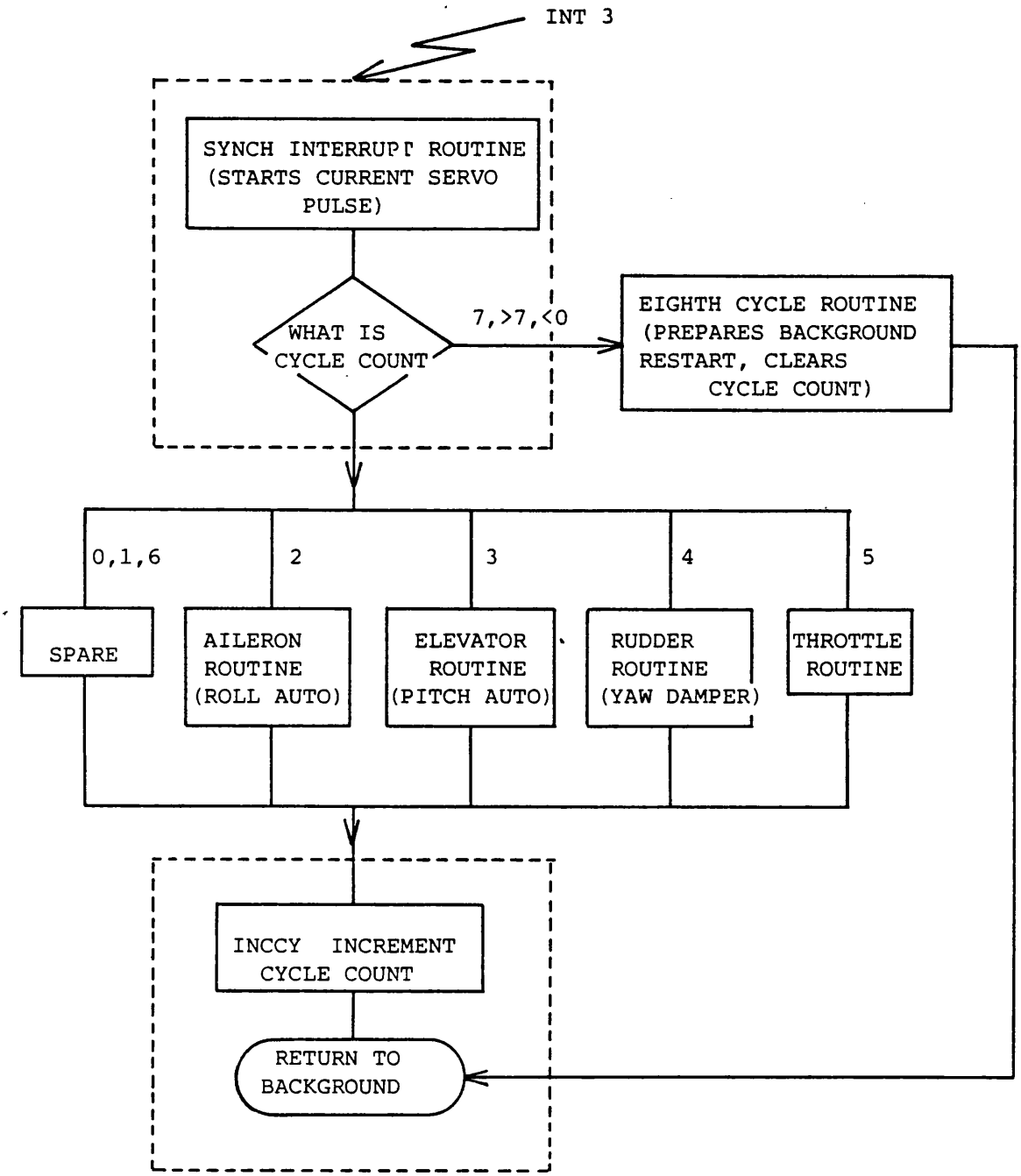


FIG. 28 OVERALL TIMING DIAGRAM. (C.P.U. USAGE)



PARTS WITHIN DOTTED LINES

ARE SERVO INTERRUPT ROUTINE.

FIG. 29 FLOW DIAGRAM OF THE FOREGROUND SOFTWARE.

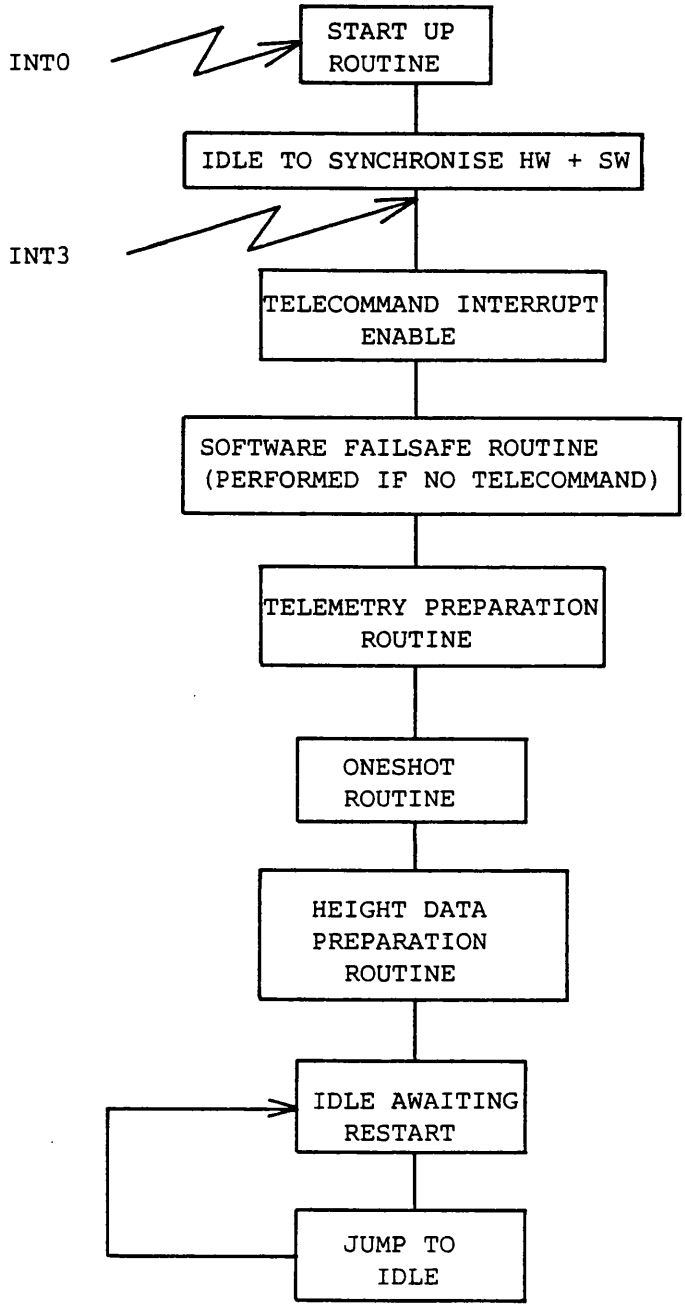
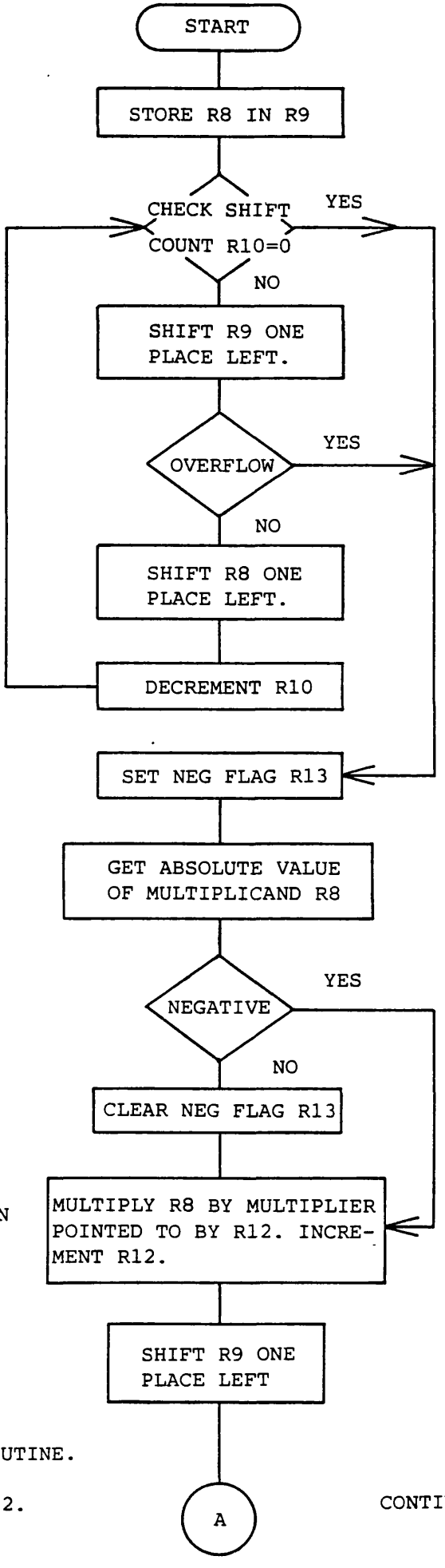


FIG.30 FLOW DIAGRAM OF THE BACKGROUND SOFTWARE.

MULT

R8 is Multiplicand, R10 is shift count, R11 is return reg.
R12 is Multiplier pointer



32 BIT RESULT IS IN R8 AND R9 L S WORD

FIG. 31 MULTIPLY ROUTINE.

SHEET 1 OF 2.

CONTINUED OVER.

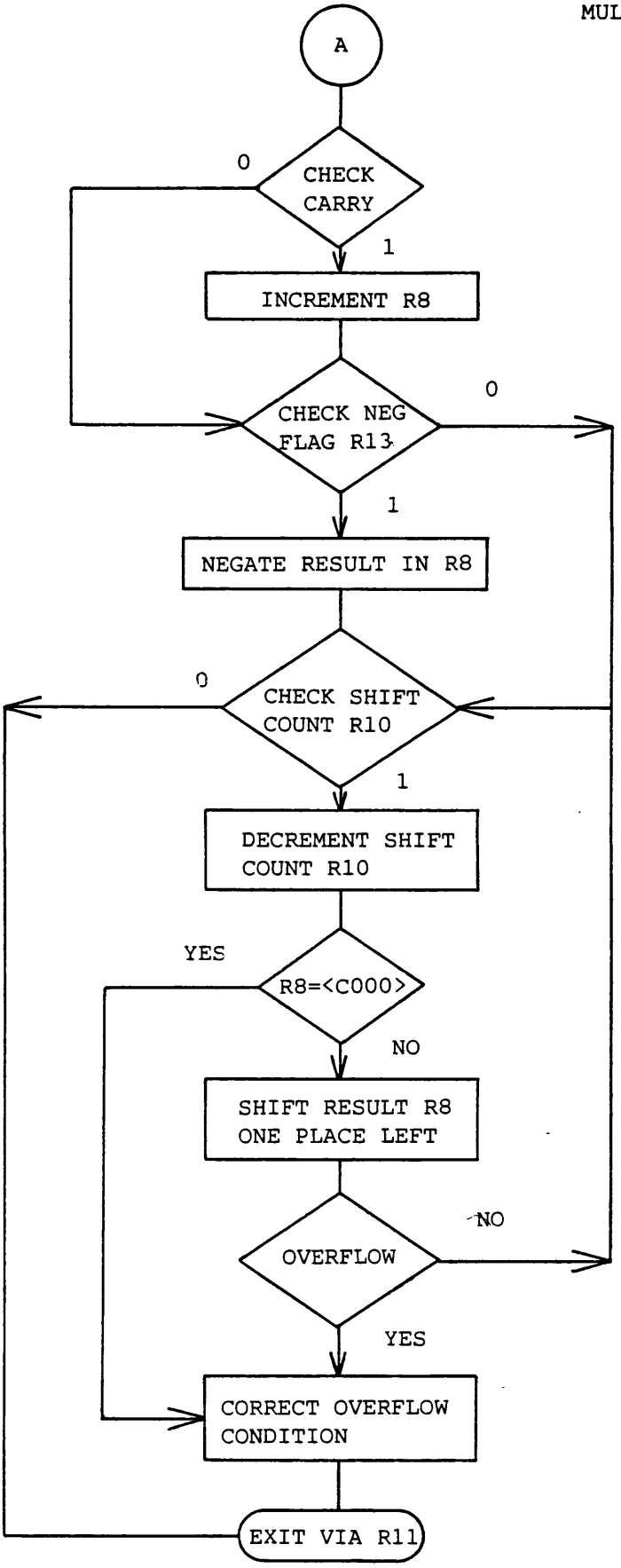


FIG.31 MULTIPLY ROUTINE.

SENS

R8 contains the no. of times the channel is to be read

R10 contains the channel to be measured.

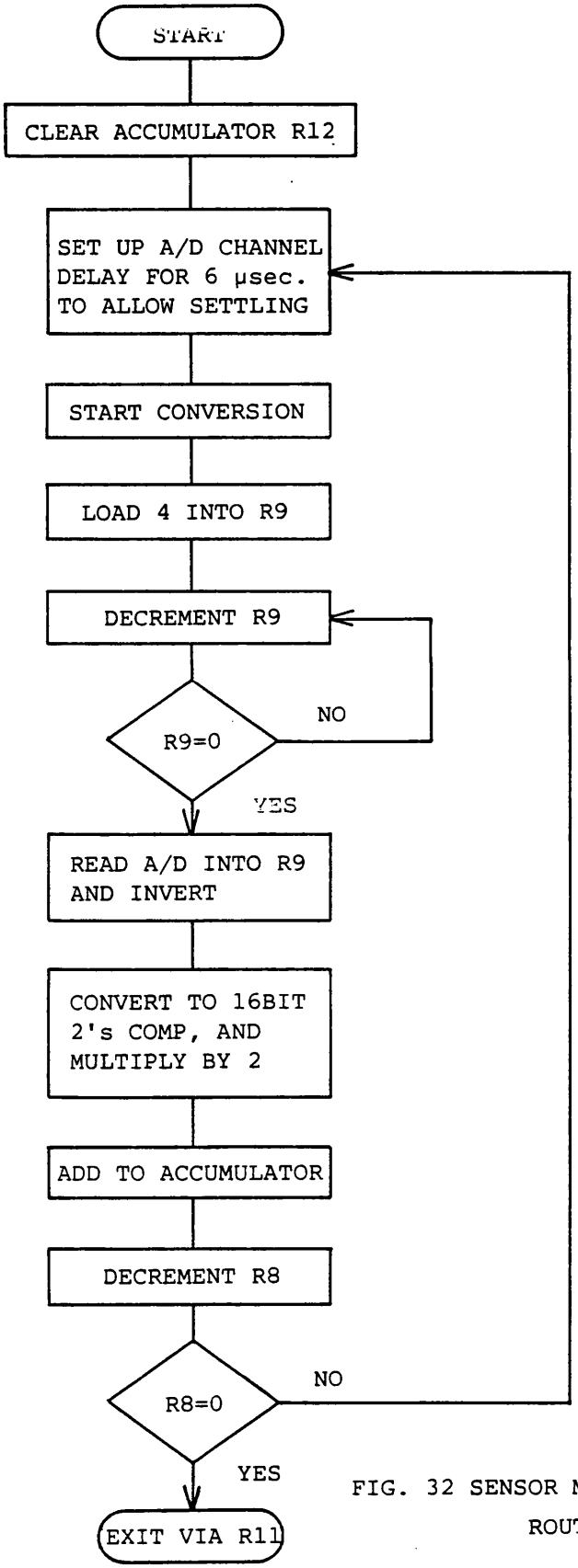


FIG. 32 SENSOR MEASUREMENT ROUTINE.

SATAD

Add two's complement no.s in R7 and R8 SCALE Data in R13

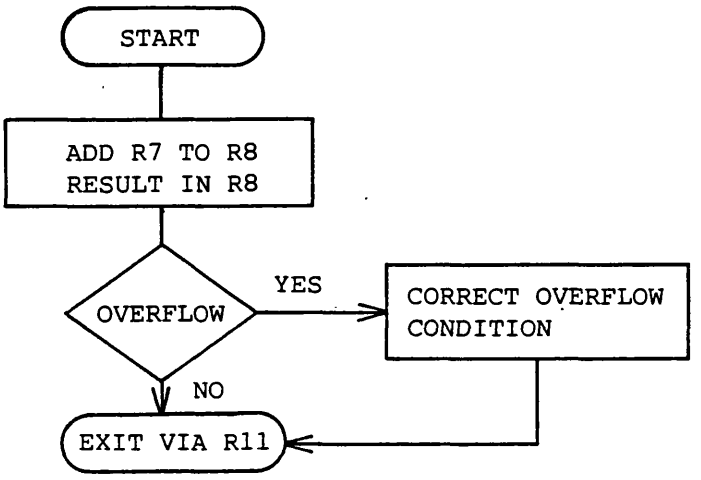


FIG. 33 SATURATION ADDING ROUTINE.

DATA IN R13 SCALE

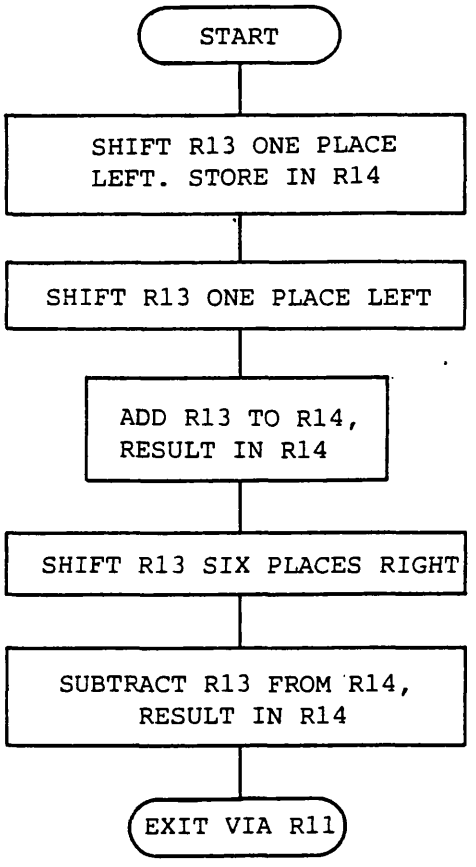


FIG. 34 PILOT MODE OUTPUT SCALING ROUTINE.

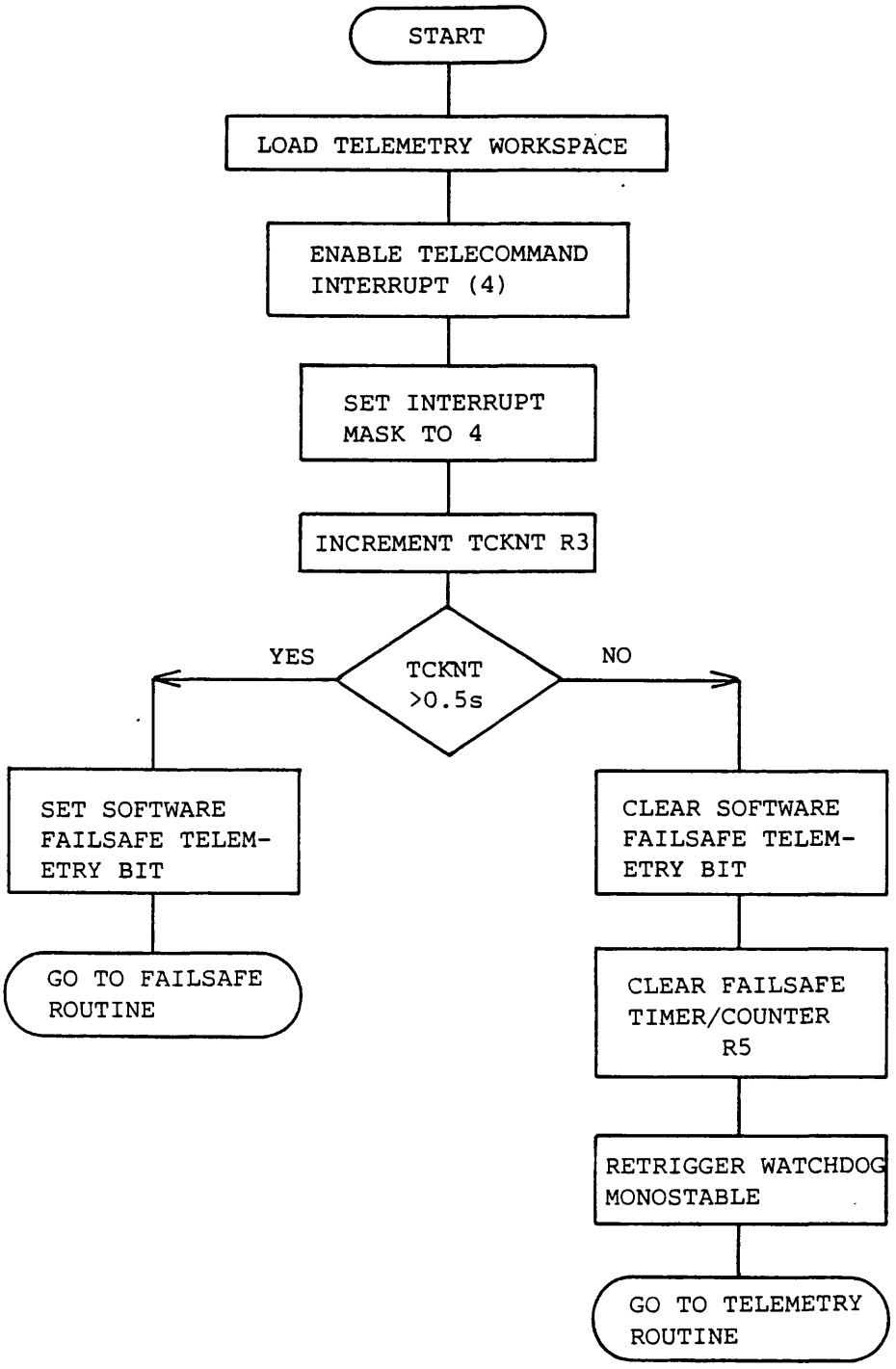


FIG. 35 TELECOMMAND INTERRUPT ENABLE ROUTINE.

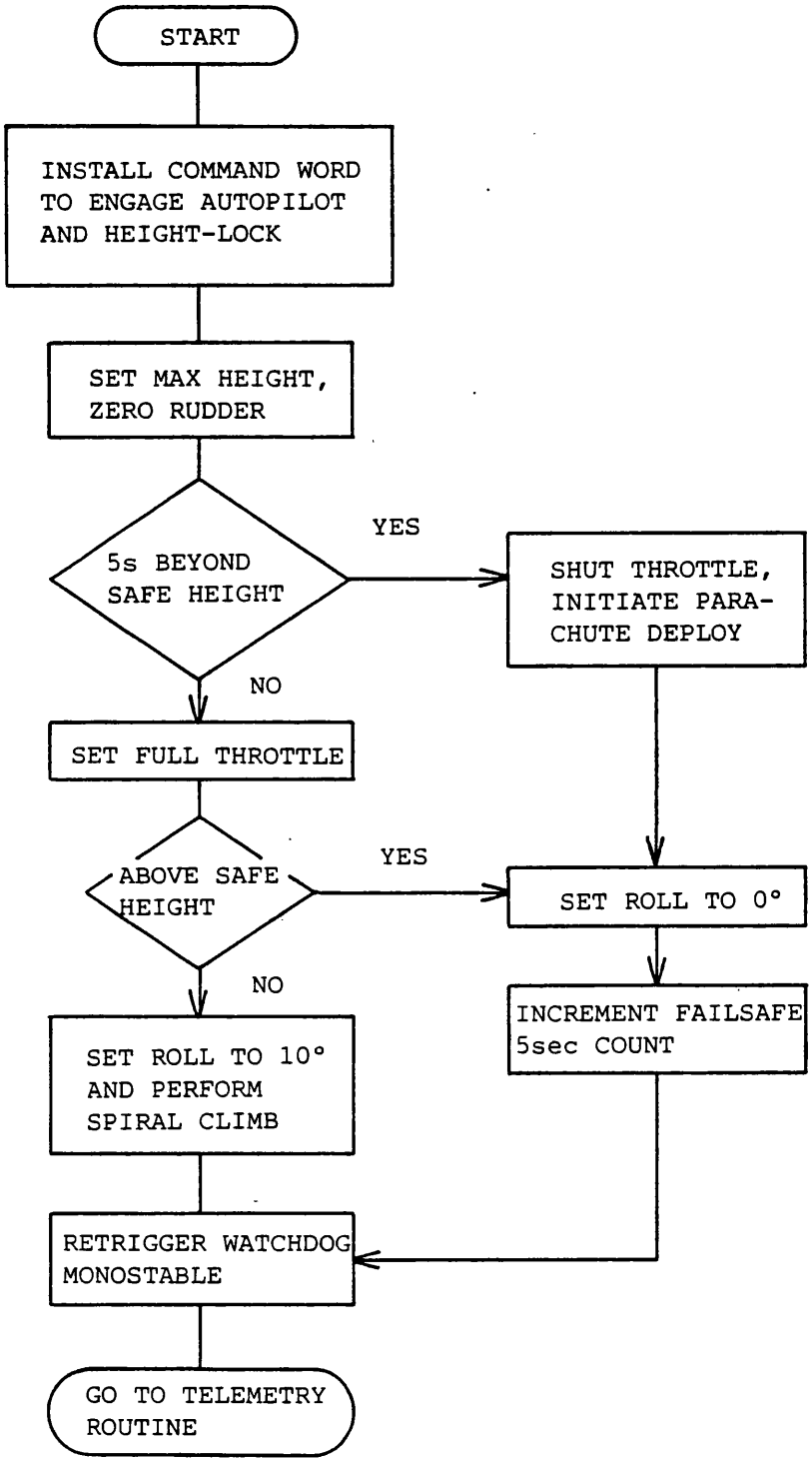


FIG.36 SOFTWARE FAILSAFE ROUTINE.

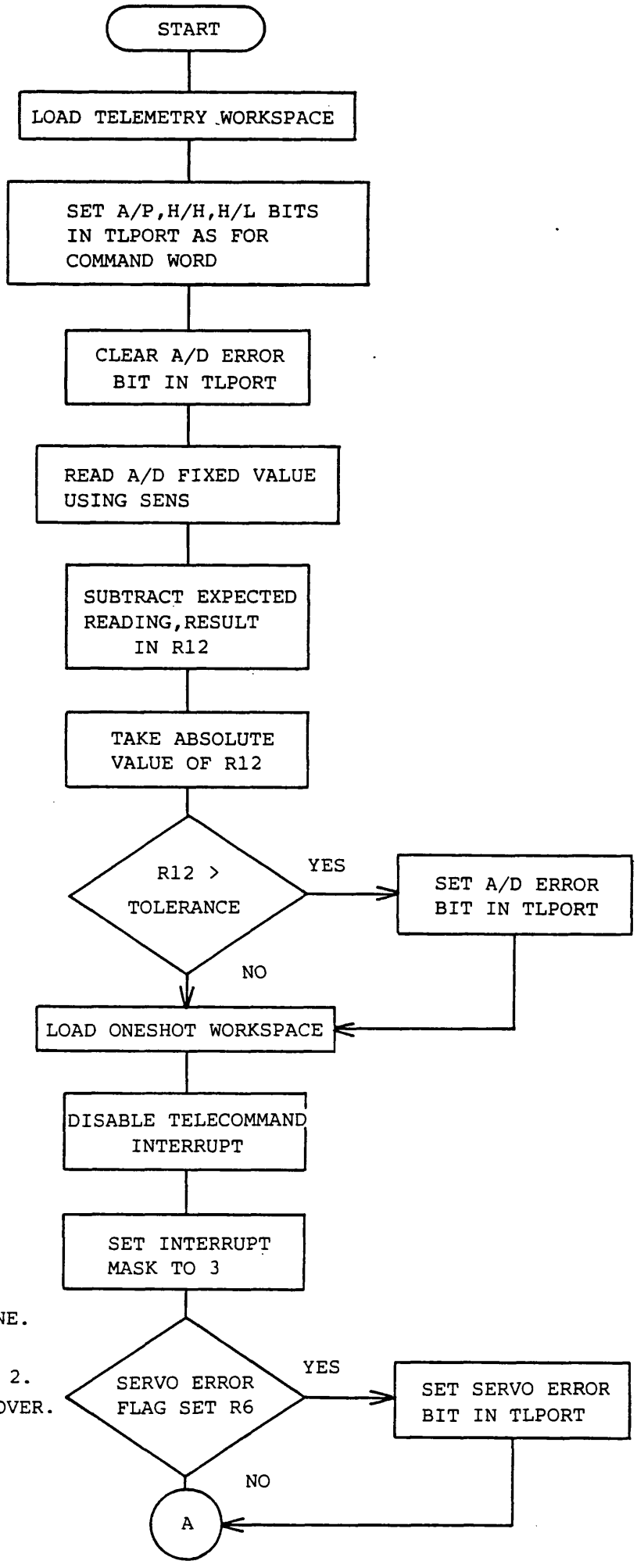


FIG. 37 TELEMETRY PREPARATION ROUTINE.

SHEET 1 OF 2.

CONTINUED OVER.

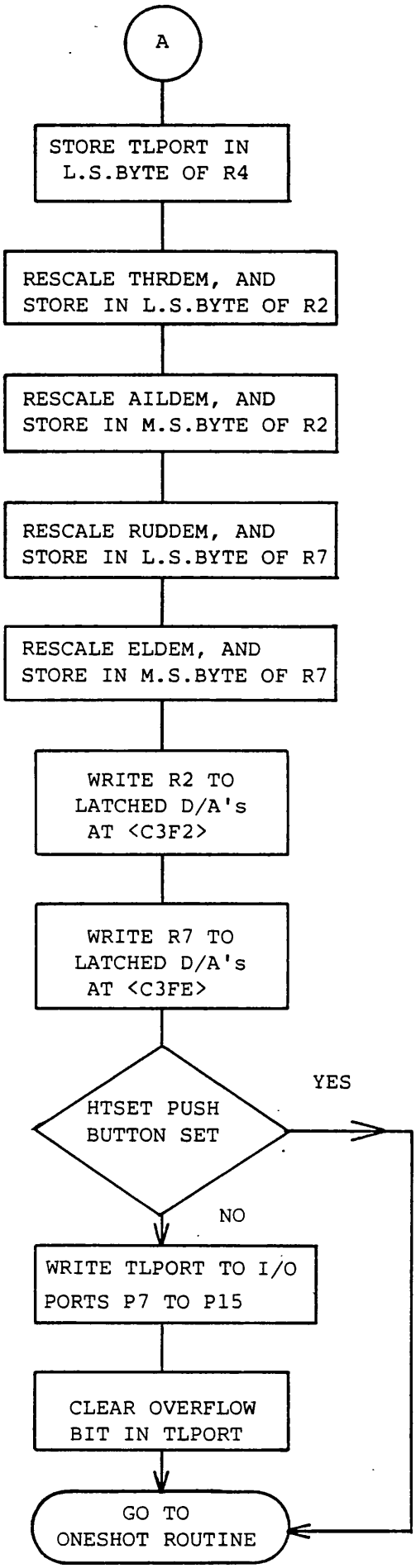


FIG. 37 TELEMETRY PREPARATION ROUTINE.

SHEET 2 OF 2.

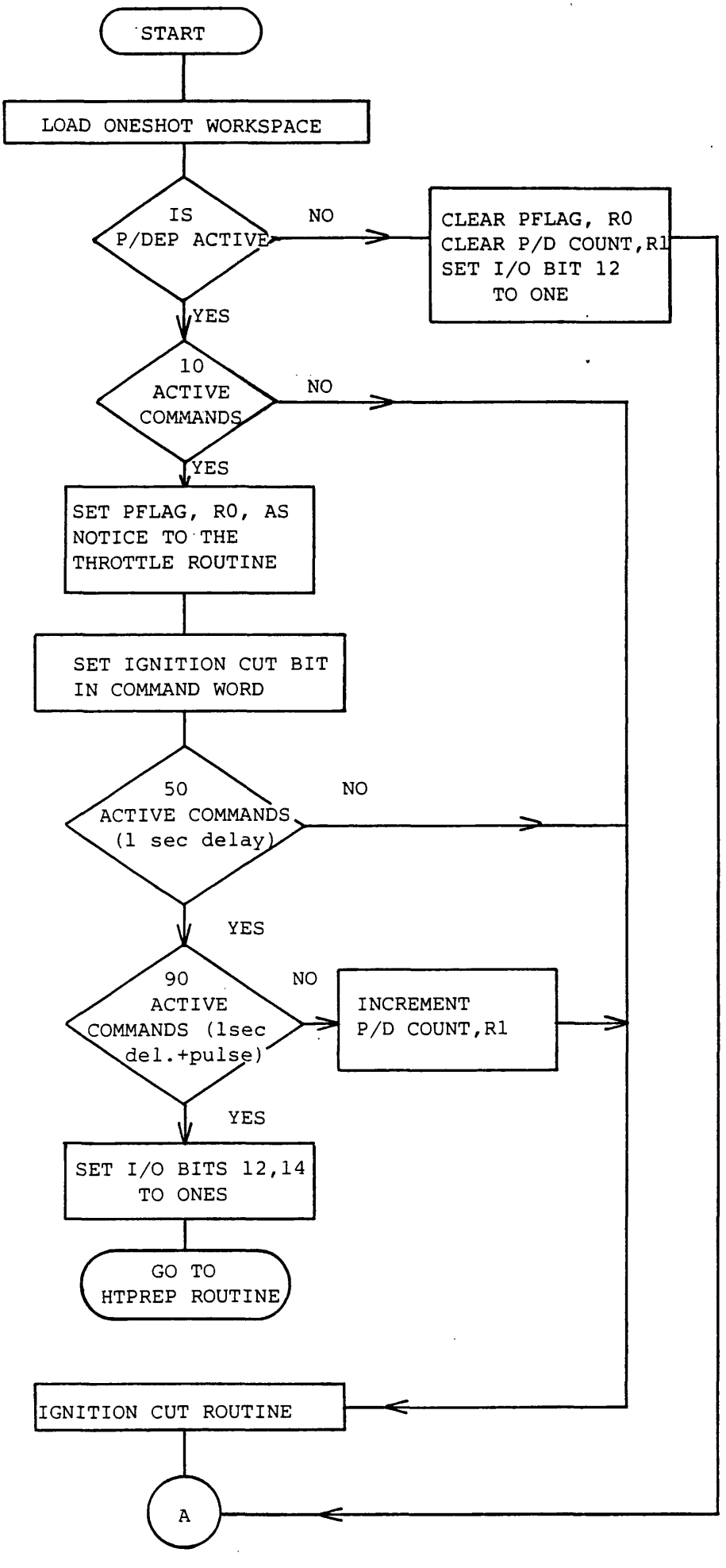


FIG. 38
THE ONESHOT ROUTINES.
SHEET 1 OF 2
CONTINUED OVER.

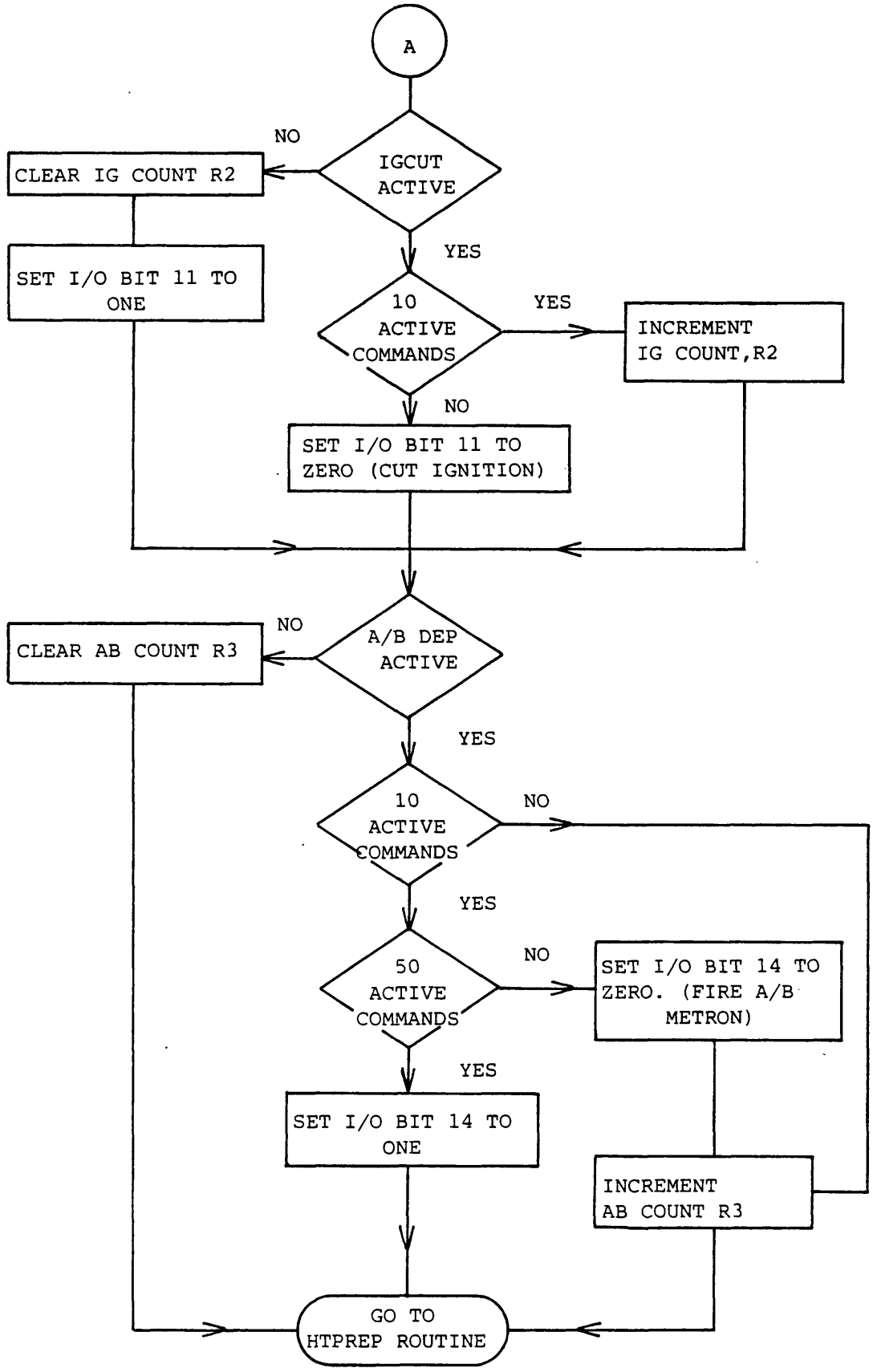


FIG. 38 THE ONESHOT ROUTINE. SHEET 2 OF 2.

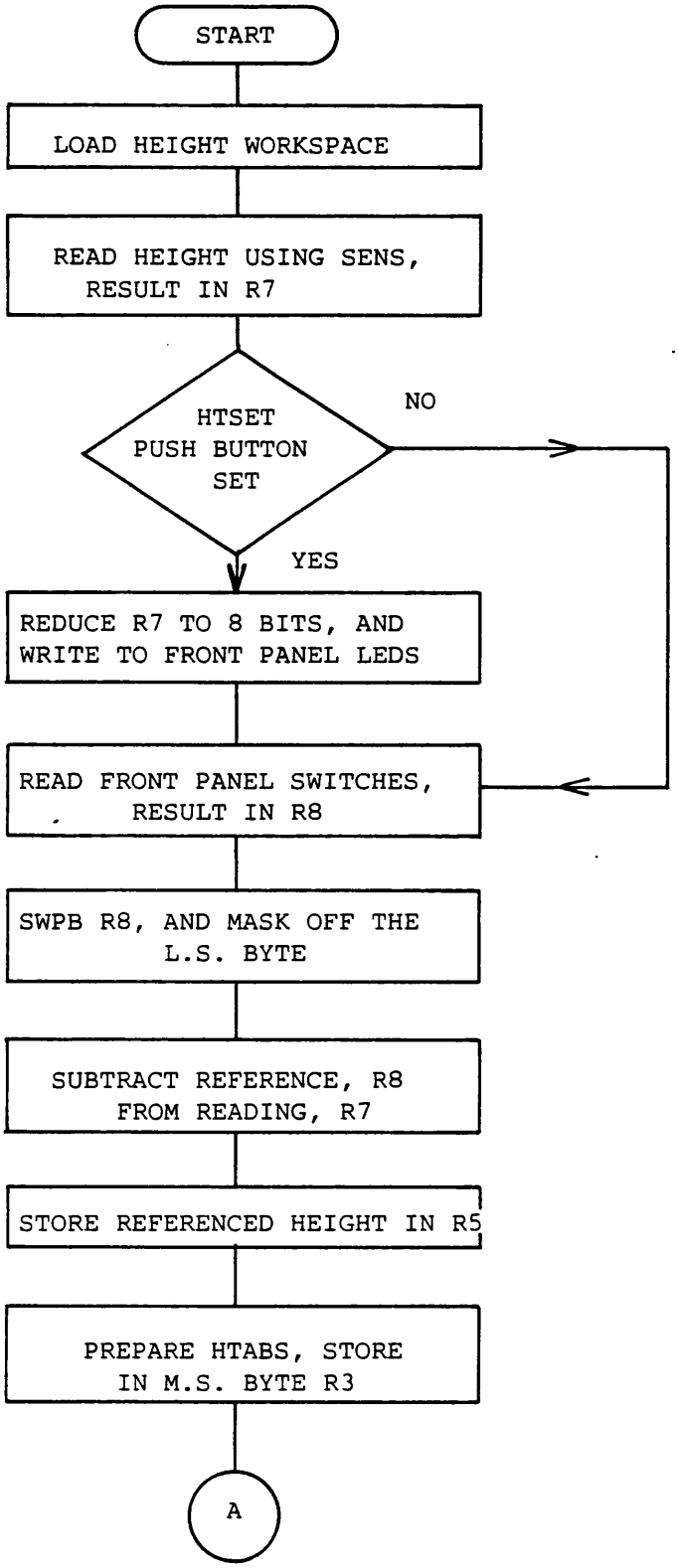


FIG. 39 HEIGHT DATA PREPARATION ROUTINE.

SHEET 1 OF 2. CONTINUED OVER.

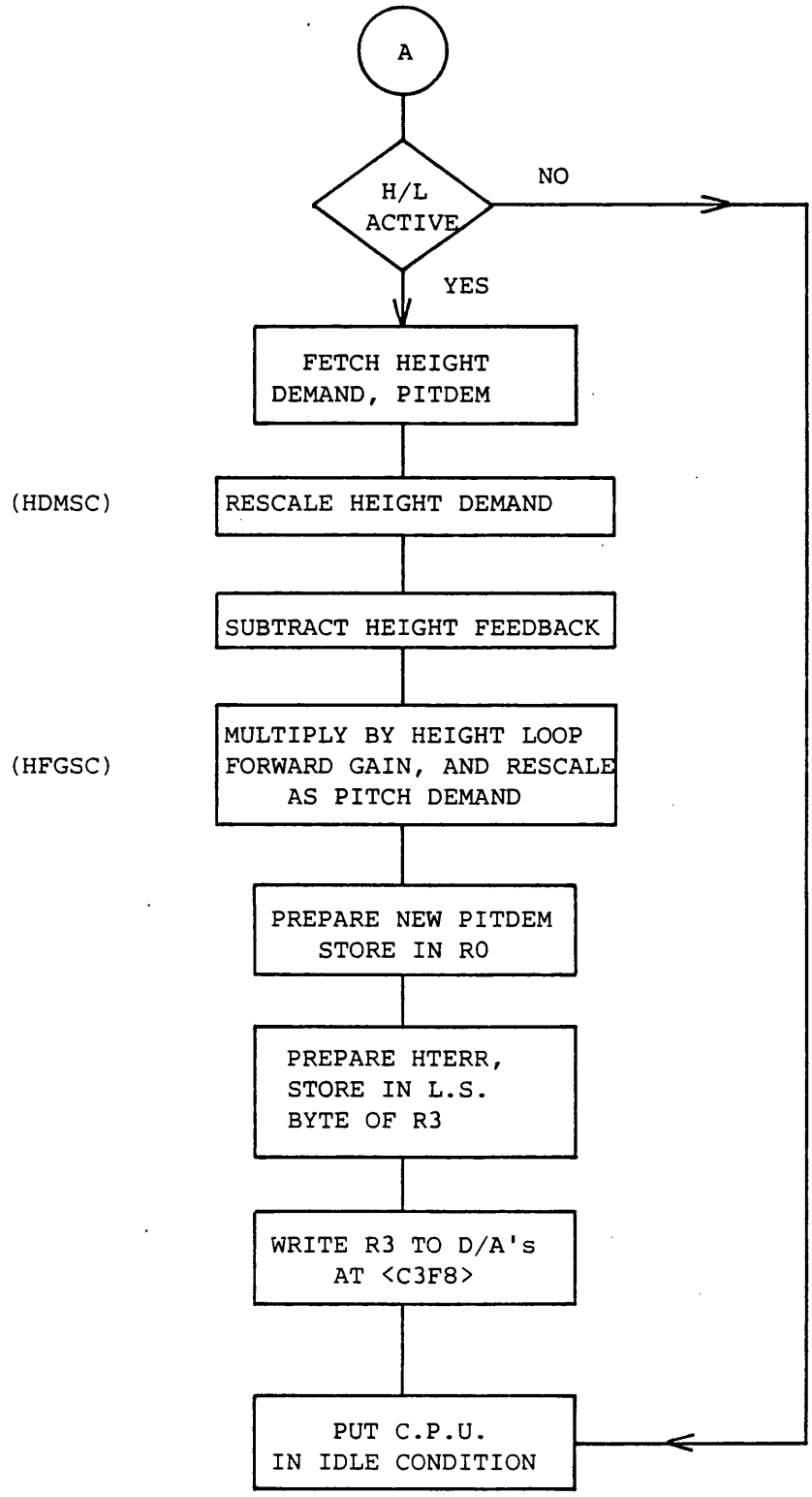


FIG. 39 HEIGHT DATA PREPARATION ROUTINE.

SHEET 2 OF 2.

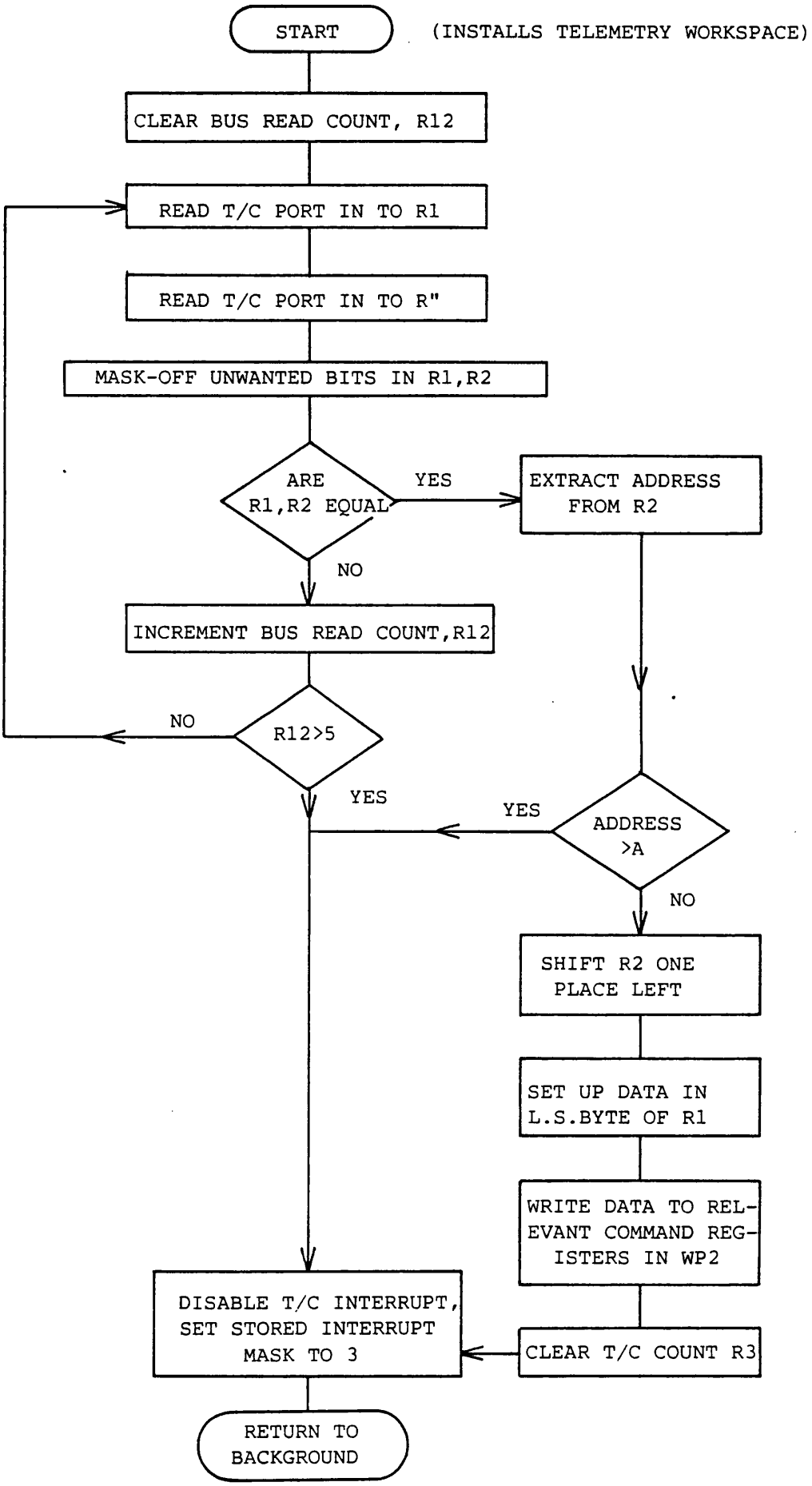
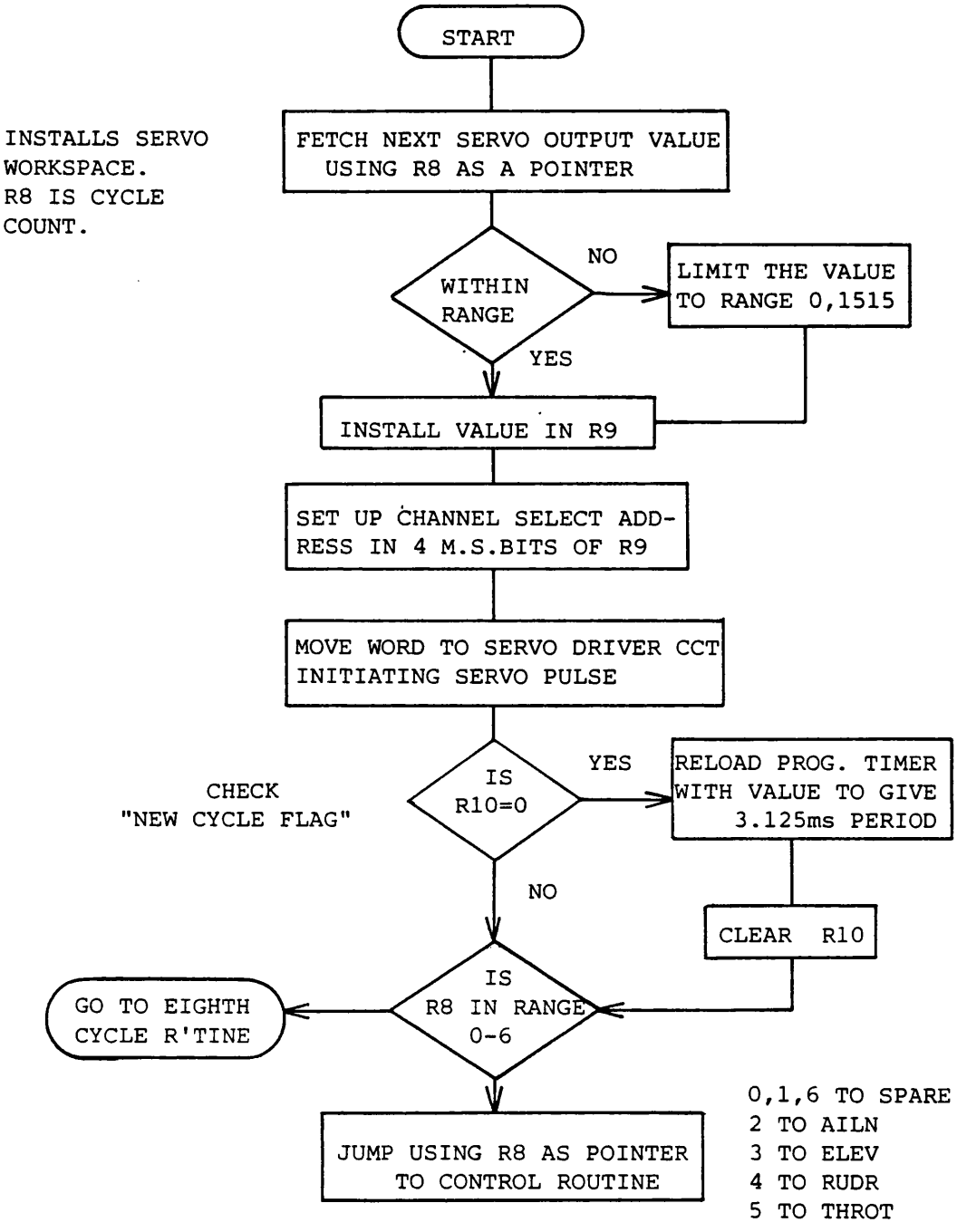


FIG. 40 INT4, THE TELECOMMAND INTERRUPT ROUTINE.



ALL CONTROL ROUTINES RETURN TO INCCY,

EIGHTH CYCLE ROUTINE RETURNS DIRECTLY

TO BACKGROUND (THE IDLE CONDITION)

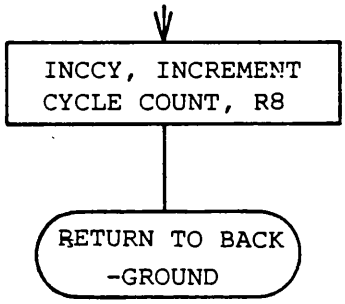


FIG. 41 THE SERVO INTERRUPT ROUTINE.

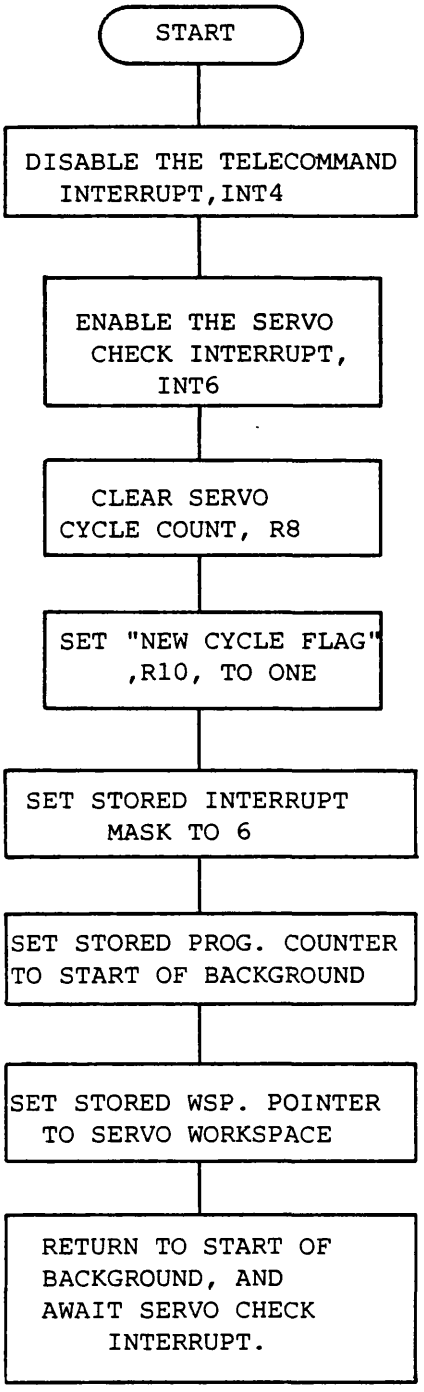


FIG. 42 THE EIGHTH CYCLE ROUTINE.

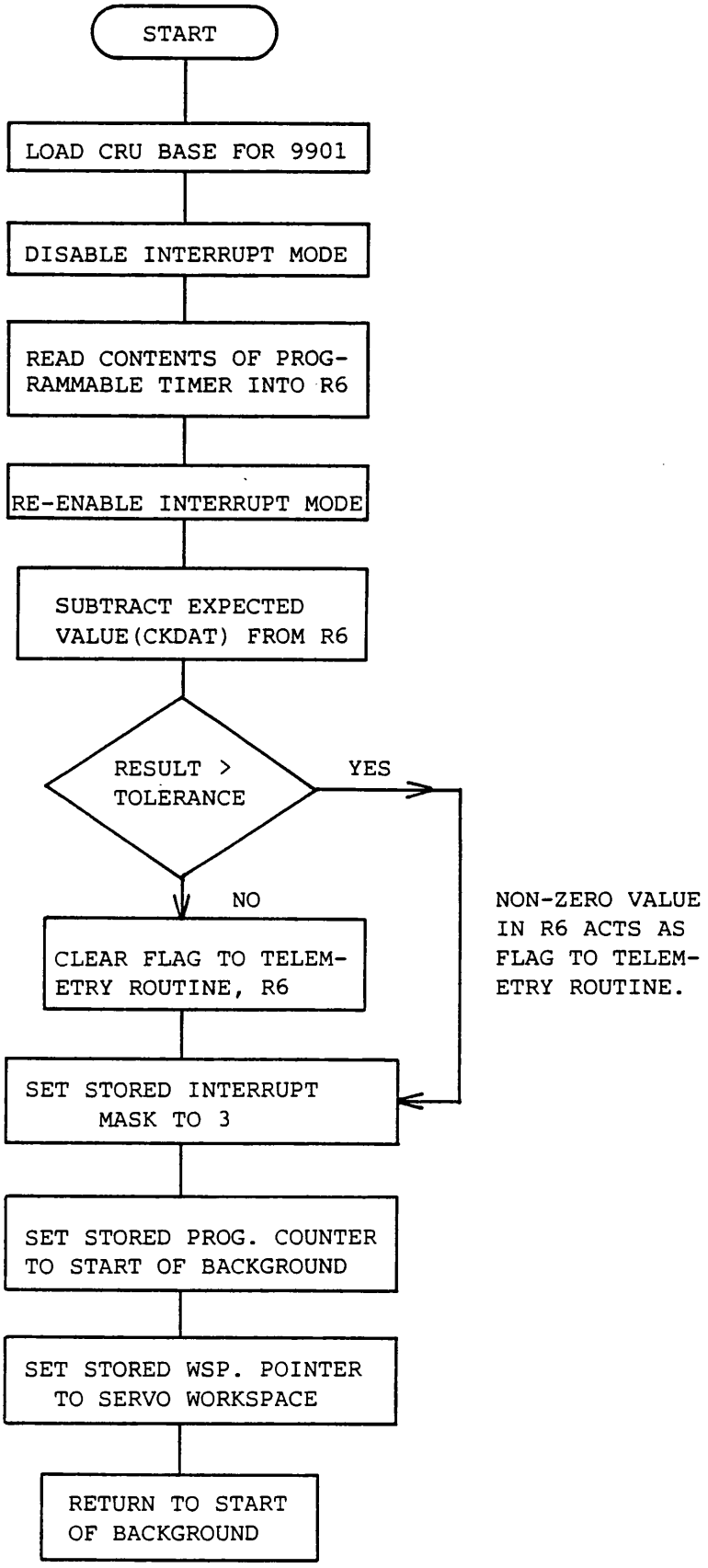


FIG. 43 INT6, THE SERVO CHECK INTERRUPT ROUTINE.

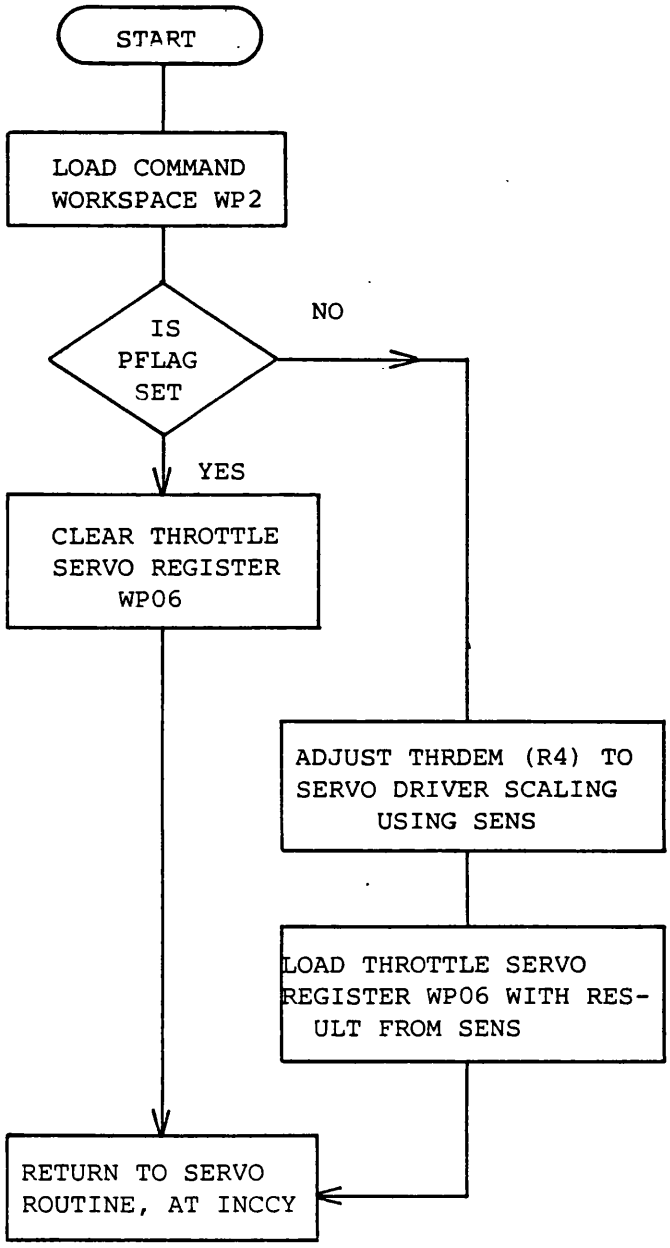


FIG. 44 THE THROTTLE ROUTINE.

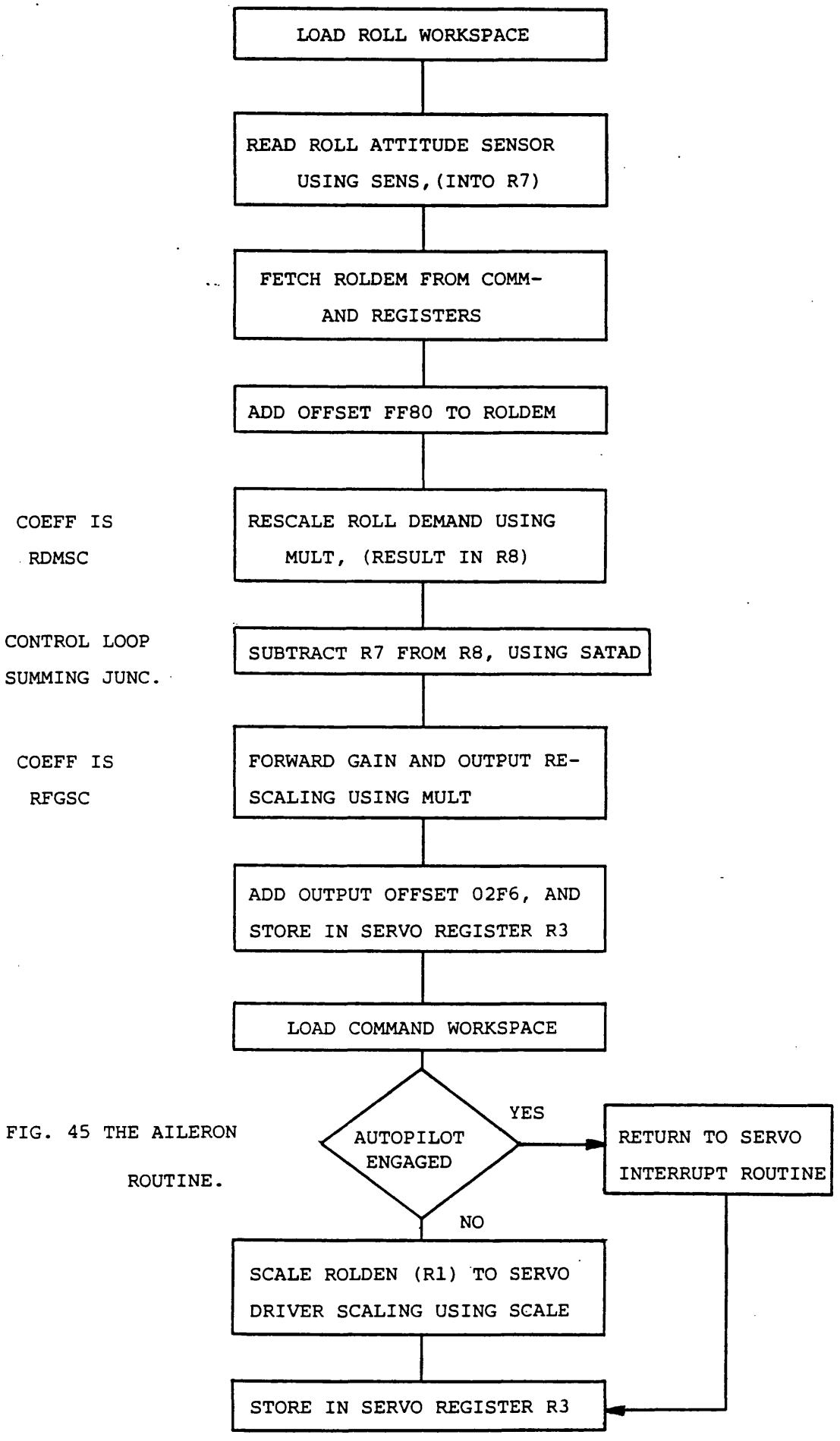


FIG. 45 THE AILERON ROUTINE.

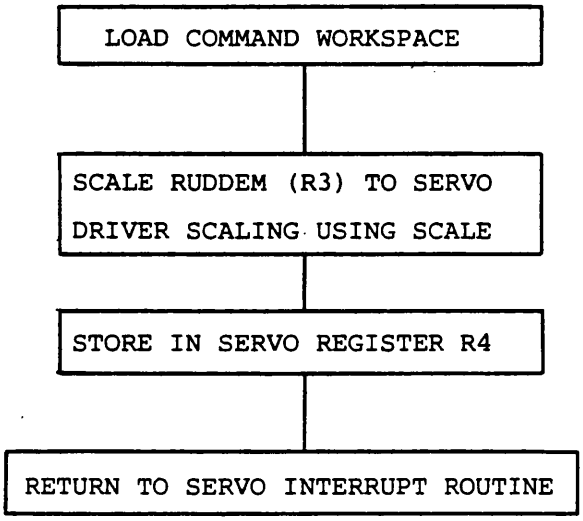


FIG. 46 THE RUDDER ROUTINE, (PILOT MODE CONTROL ONLY)

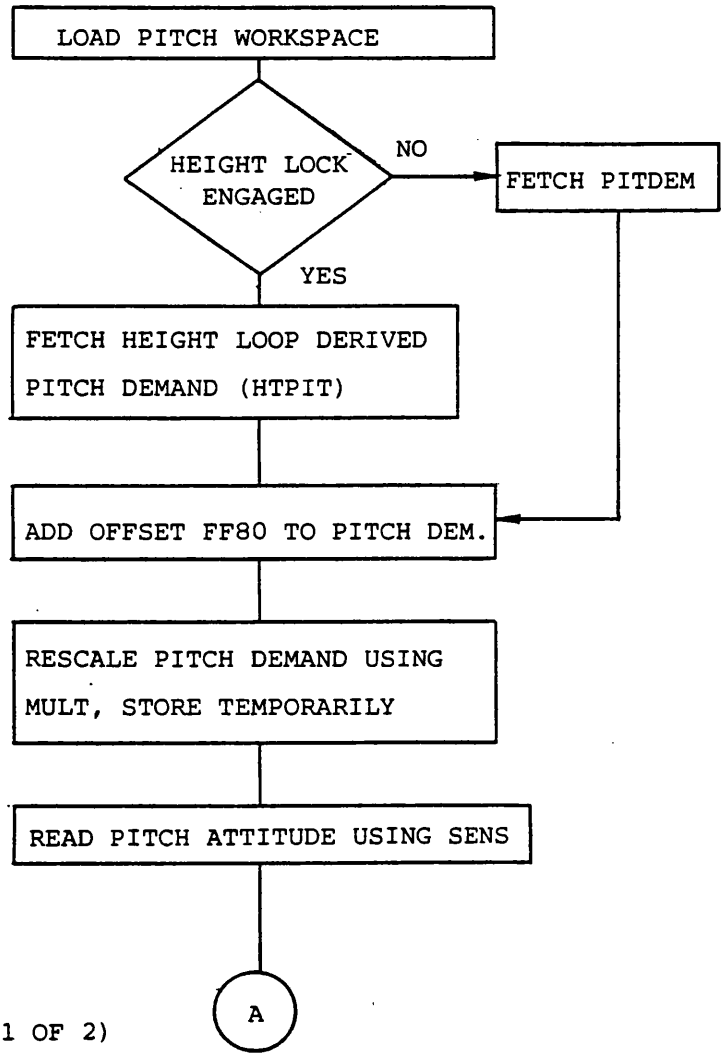


FIG. 47 THE ELEVATOR ROUTINE. (SHEET 1 OF 2)

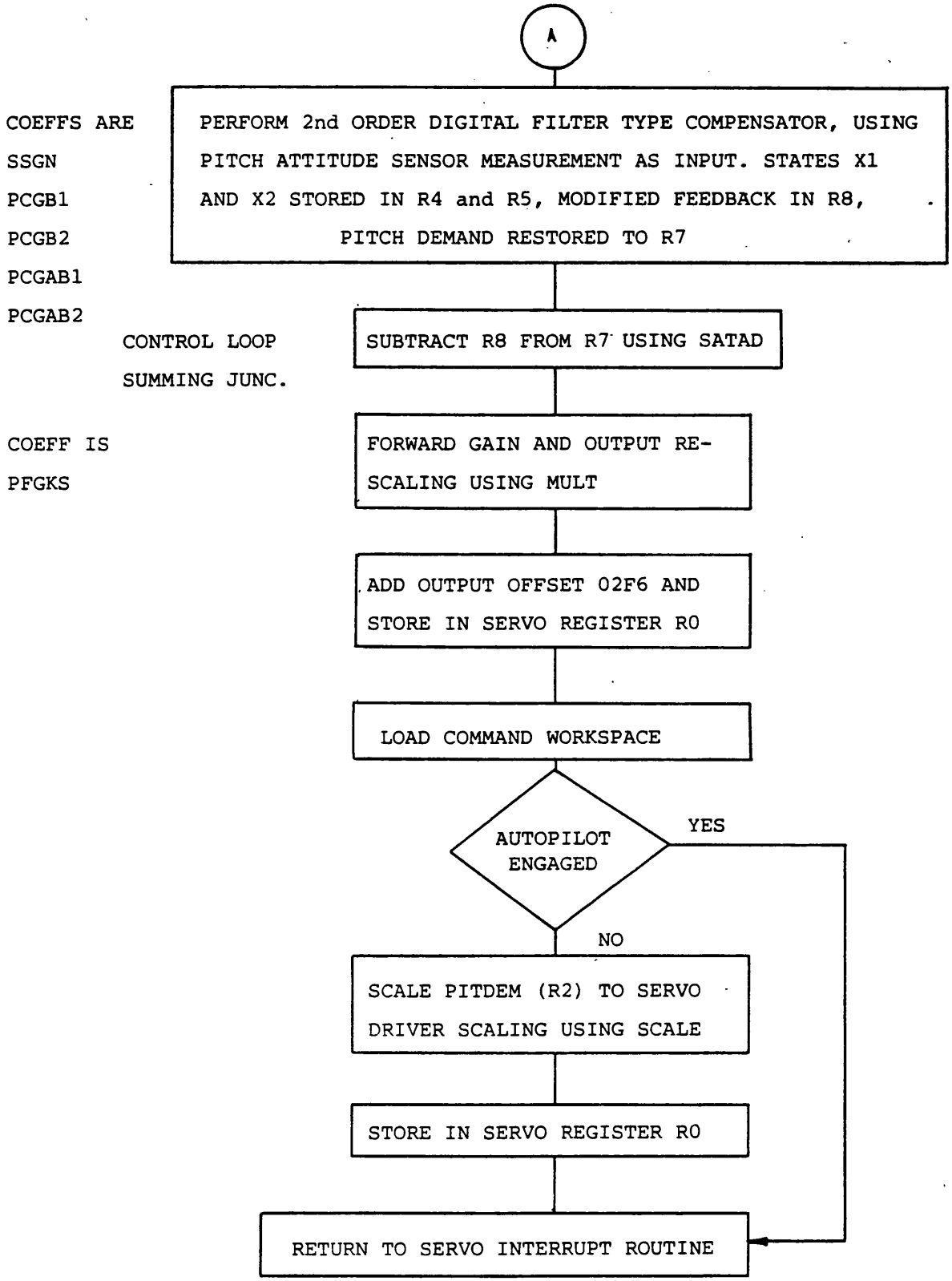


FIG. 47 THE ELEVATOR ROUTINE. (SHEET 2 OF 2).

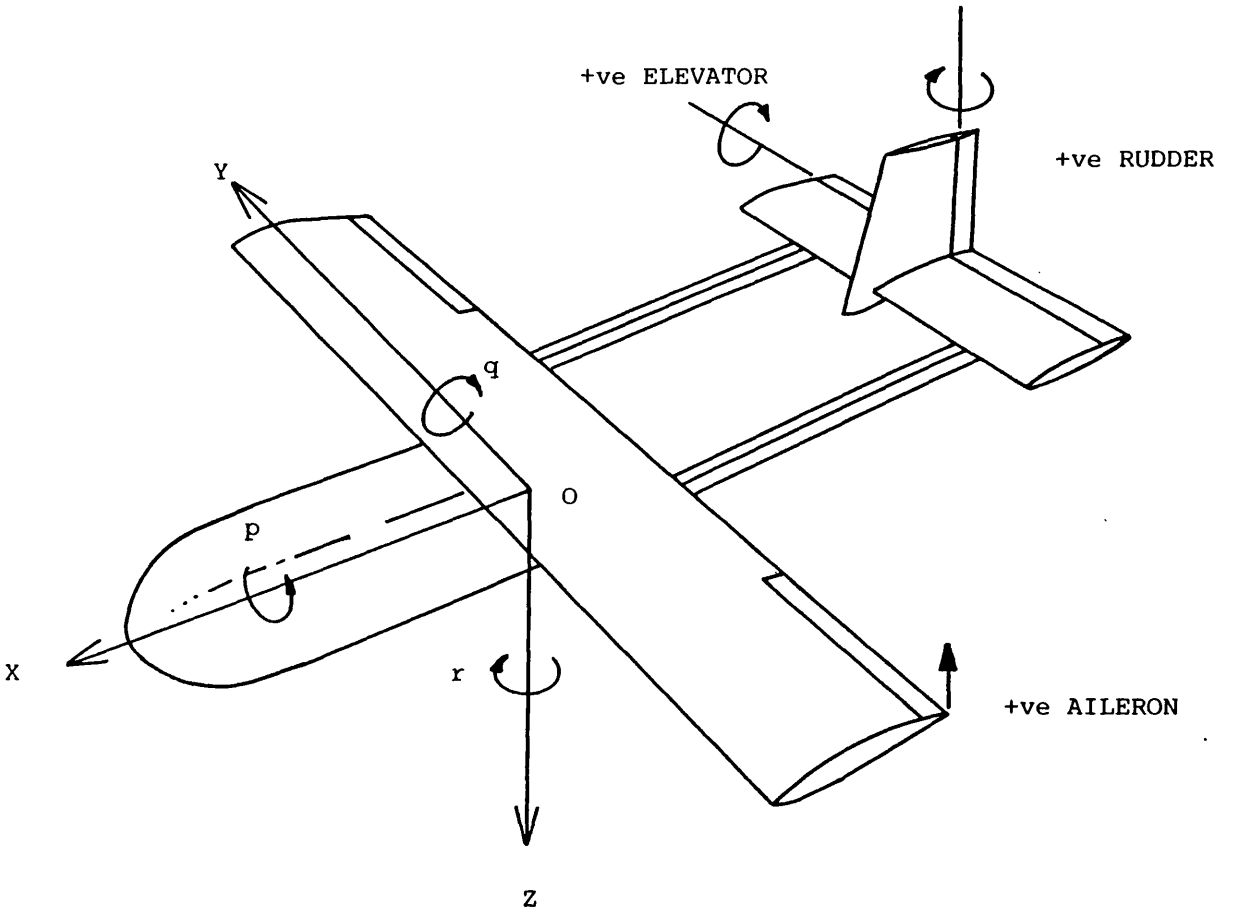


FIG.48 POSITIVE SENSES OF THE BODY AXES, THE BODY RATES AND THE CONTROL SURFACE DEFLECTIONS.

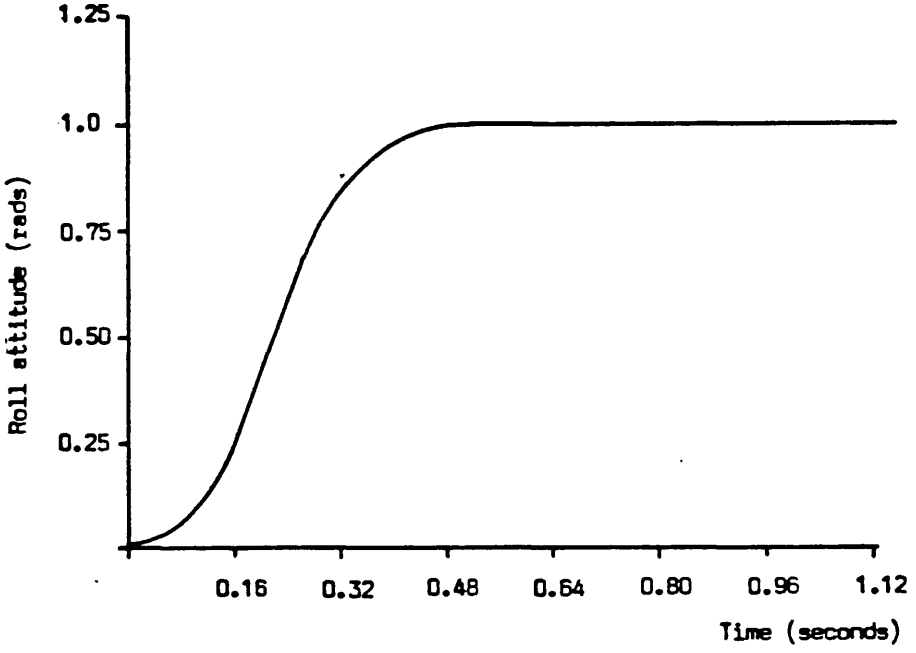


Fig. 49 Roll step response guideline (reproduced from (14))

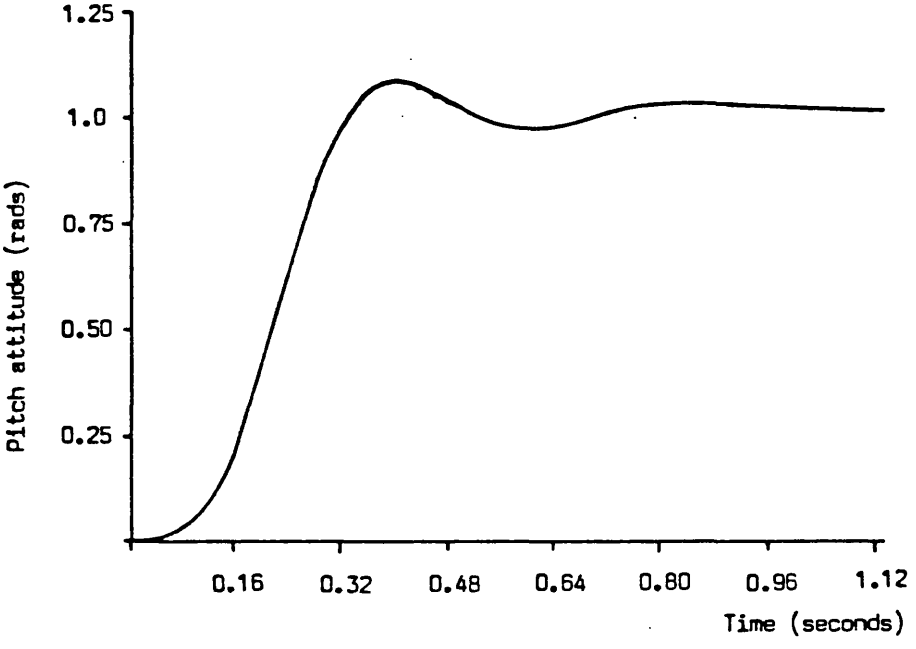
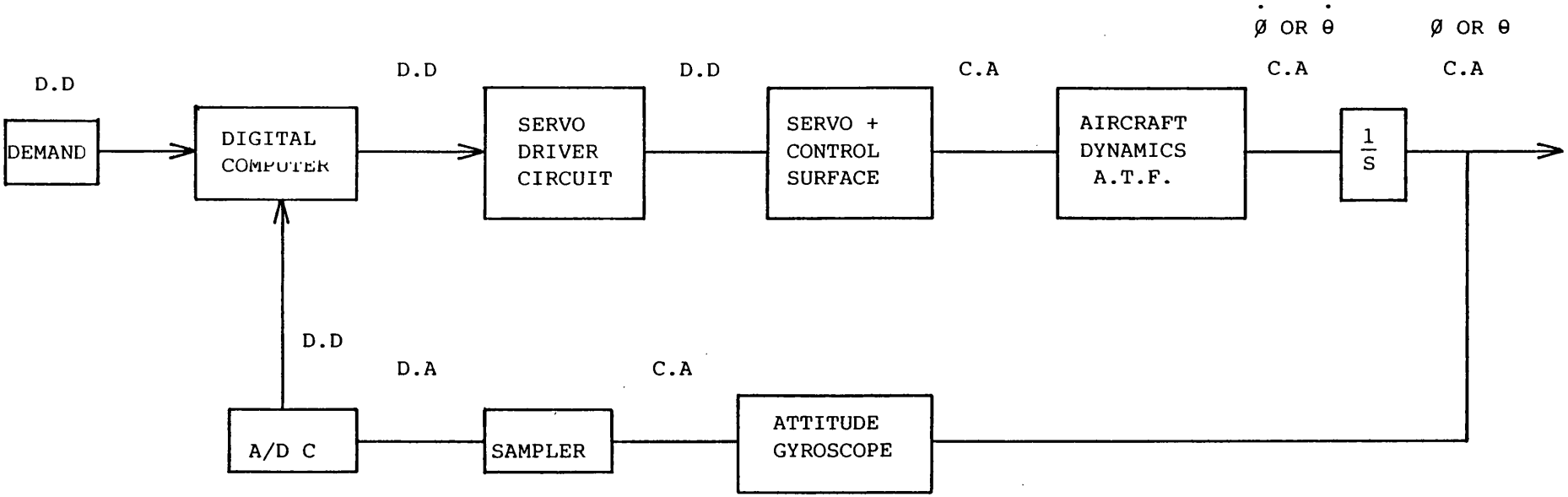


Fig. 50 Pitch step response guideline (reproduced from (14))



KEY TO SIGNAL TYPES

D.D DISCRETE, DIGITAL

D.A DISCRETE, ANALOGUE

C.A CONTINUOUS, ANALOGUE

FIG. 51 CONFIGURATION OF A DFCS AUTOPILOT LOOP.

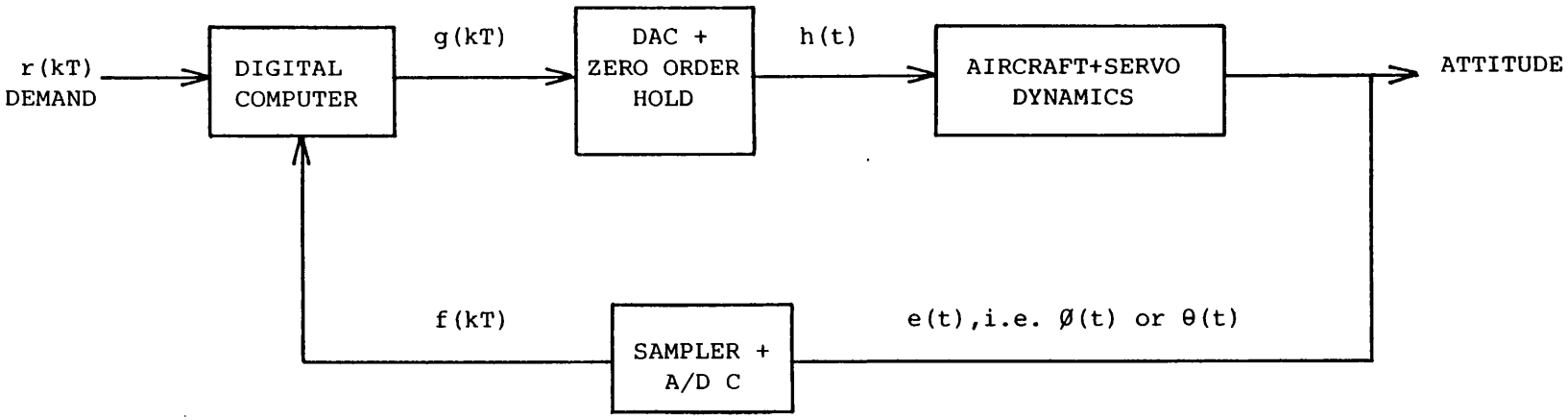


FIG. 52 MATHEMATICALLY SIGNIFICANT OPERATIONS OF A DFCS AUTOPILOT LOOP.

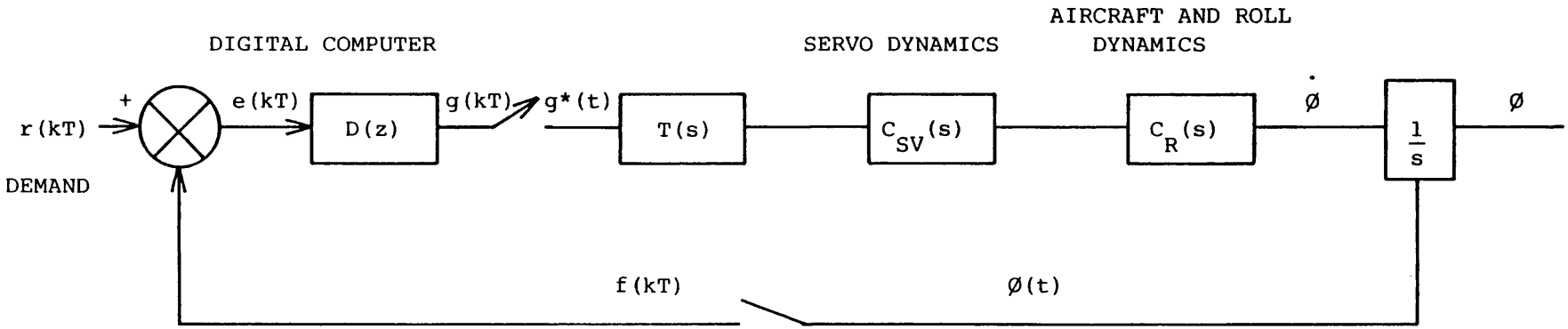


FIG. 53 THE DFCS ROLL LOOP IN TRANSFER FUNCTION FORM.

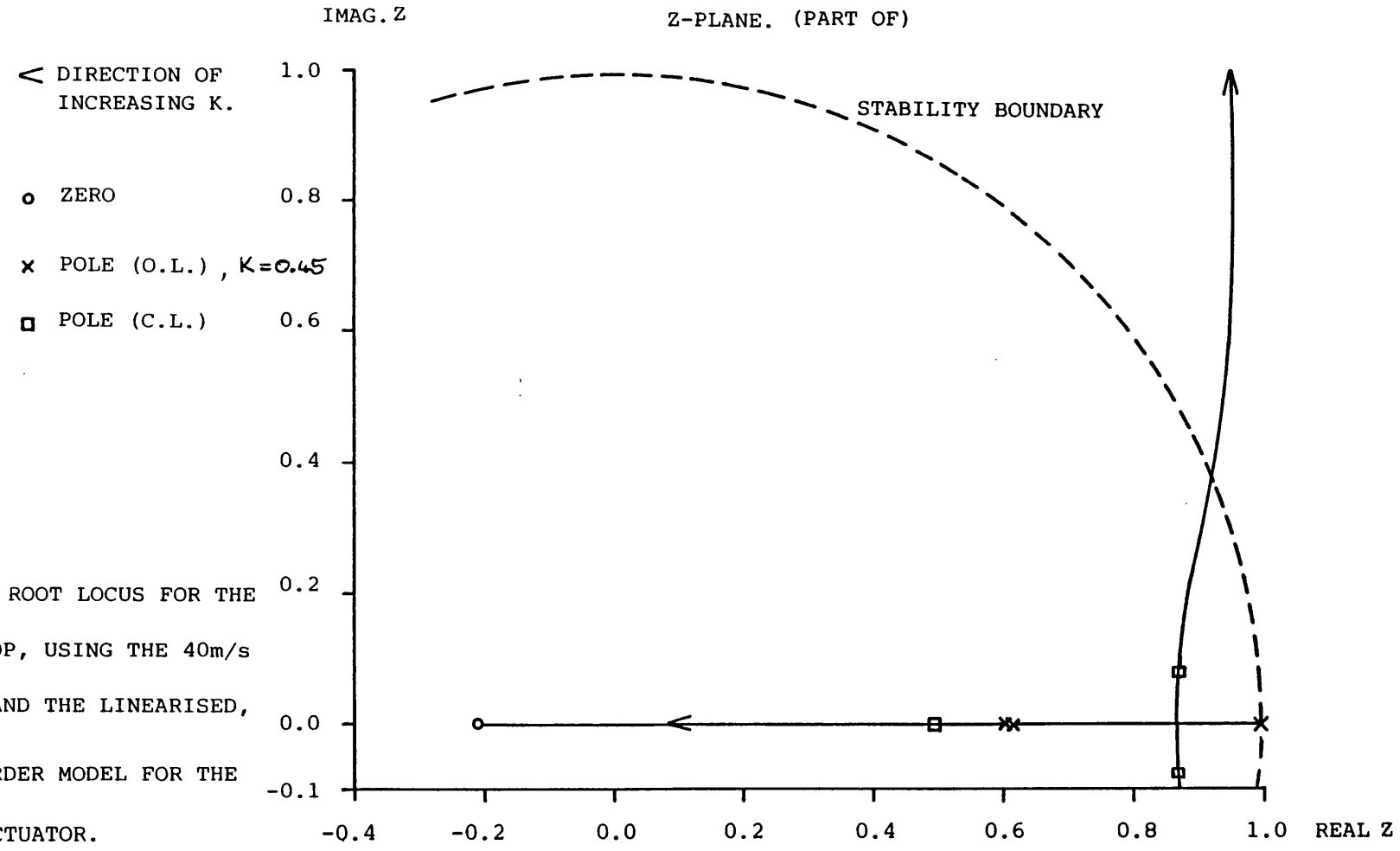


FIG. 54 ROOT LOCUS FOR THE ROLL LOOP, USING THE 40m/s A.T.F. AND THE LINEARISED, FIRST ORDER MODEL FOR THE SERVO-ACTUATOR.

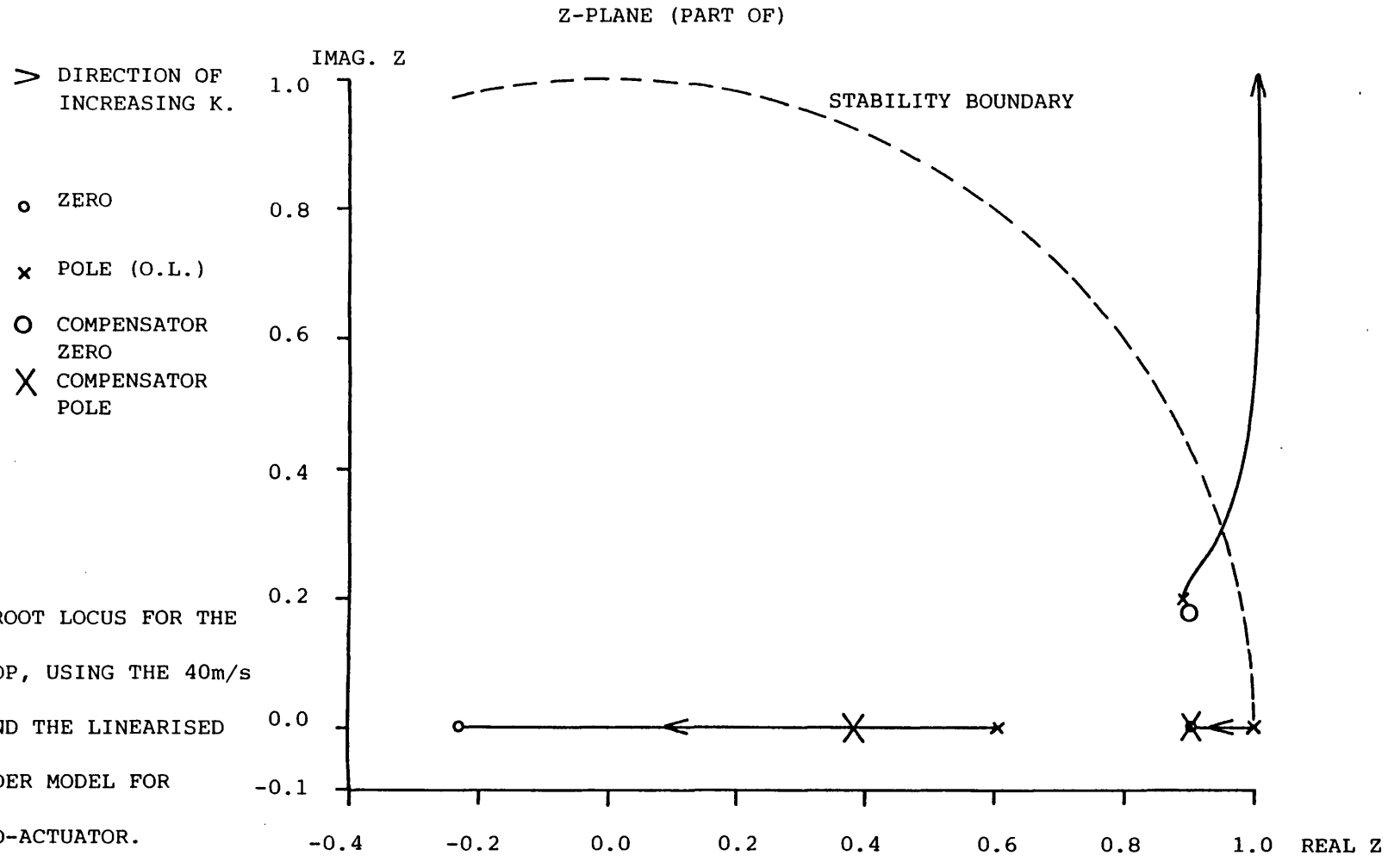


FIG. 55 ROOT LOCUS FOR THE PITCH LOOP, USING THE 40m/s A.T.F. AND THE LINEARISED FIRST ORDER MODEL FOR THE SERVO-ACTUATOR.

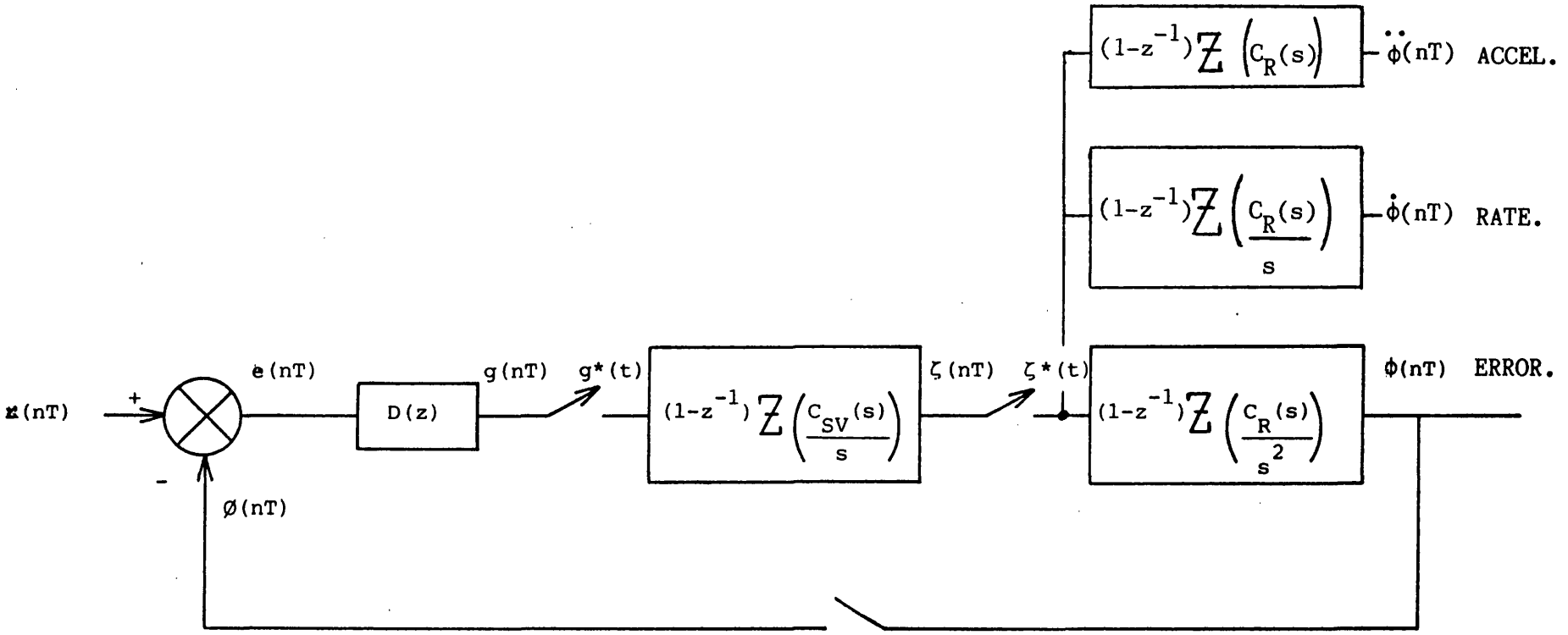


FIG. 56 TRANSFER FUNCTION DIAGRAM OF THE ROLL LOOP SIMULATION.

(Rate and acceleration are used for feedback purposes, see part III).

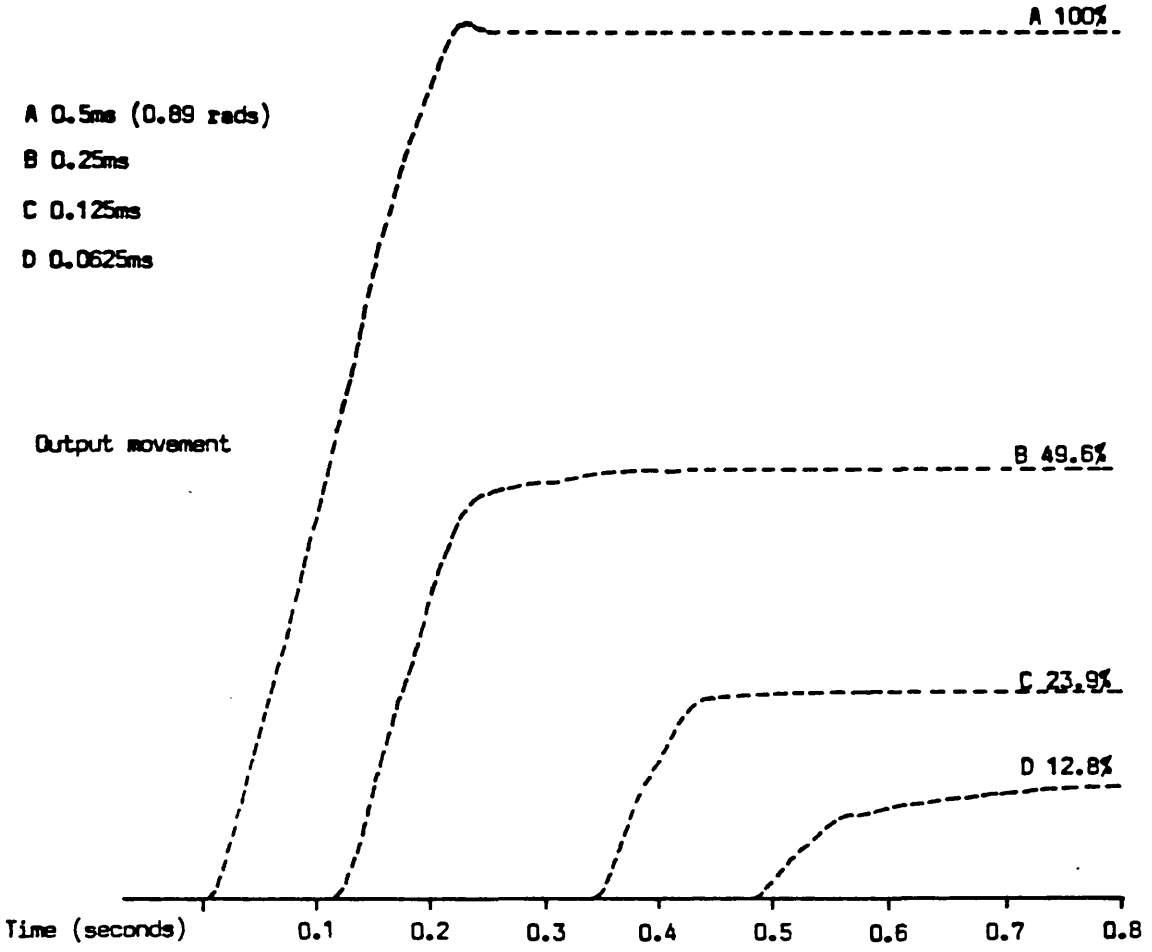


Fig. 57 Response of Skyleader SRC44B servomotor to step changes of input demand, (large differences).

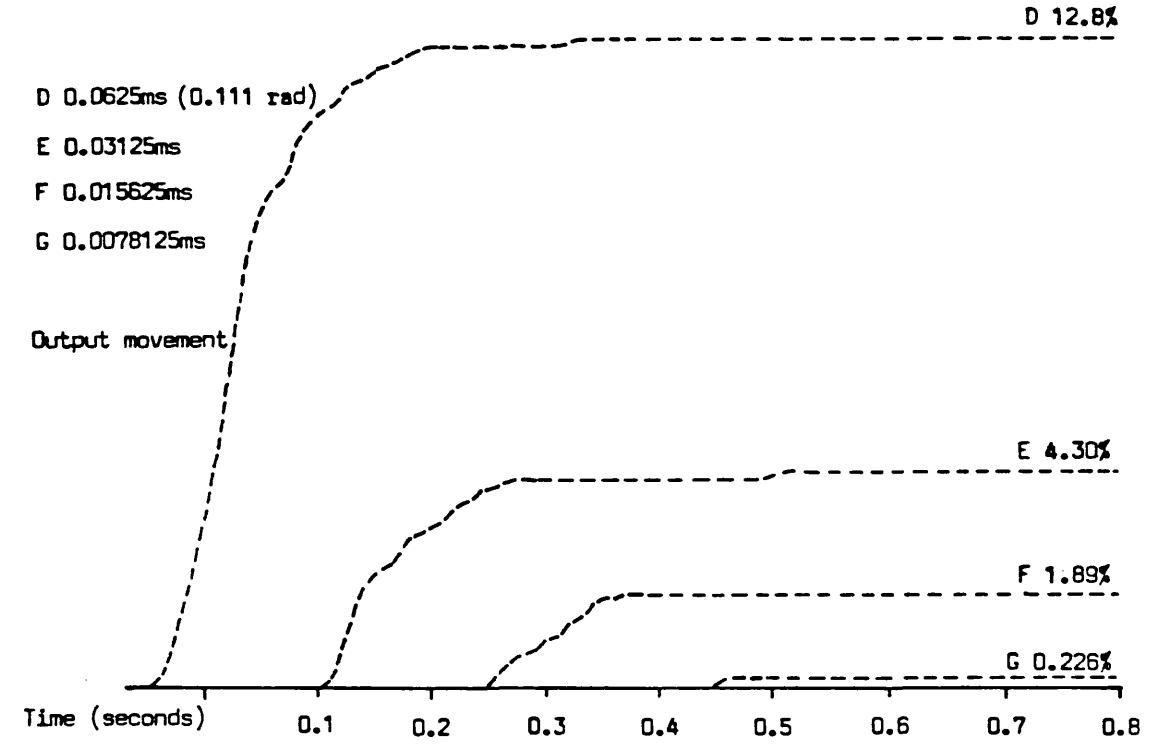


Fig. 58 Response of Skyleader SRC44B servomotor to step changes of input demand, (small differences).

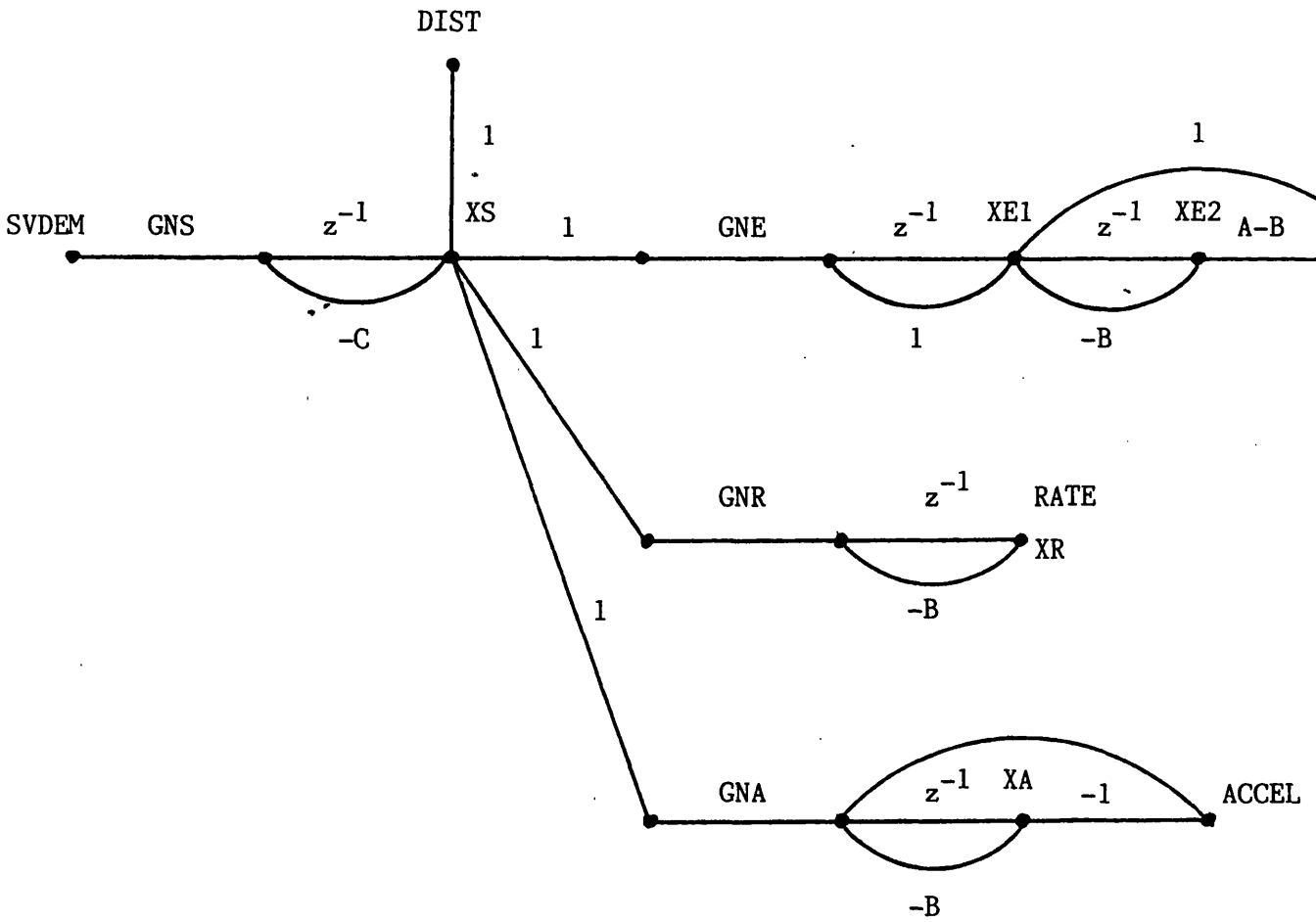


Fig. 59 State diagram of the forward path of the roll loop simulation.

(Labels refer to variable names in the programme.)

The state equations.

XS1 = GNS*SVDEM - C*XS

XE11 = GNE*(XS + DIST) + XE1

XE21 = XE1 - B*XE2

XR1 = GNR*(XS + DIST) - B*XR

XA1 = GNA*(XS + DIST) - B*XA

The output equations.

ERROR = XE1 + (A-B)*XE2

RATE = XR

ACCEL = GNA*XS - (1.0+B)*XA

Fig. 60 The difference equations.

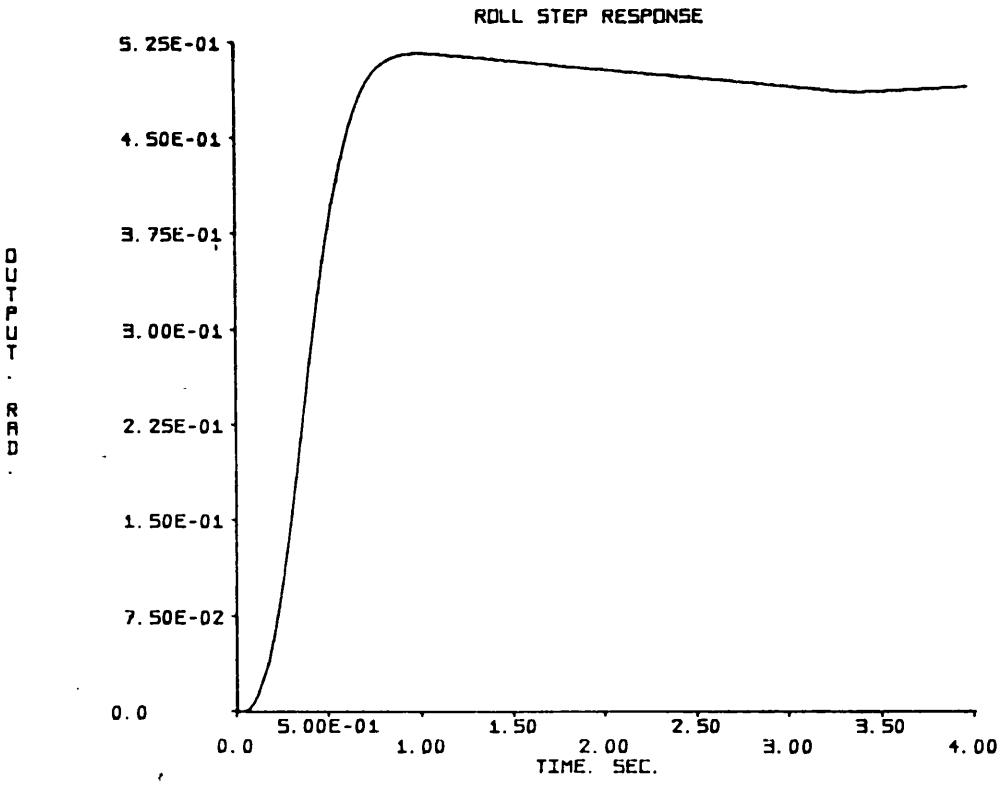


Fig. 61 Simulated roll attitude step response, 40 m/s, gain = 0.4

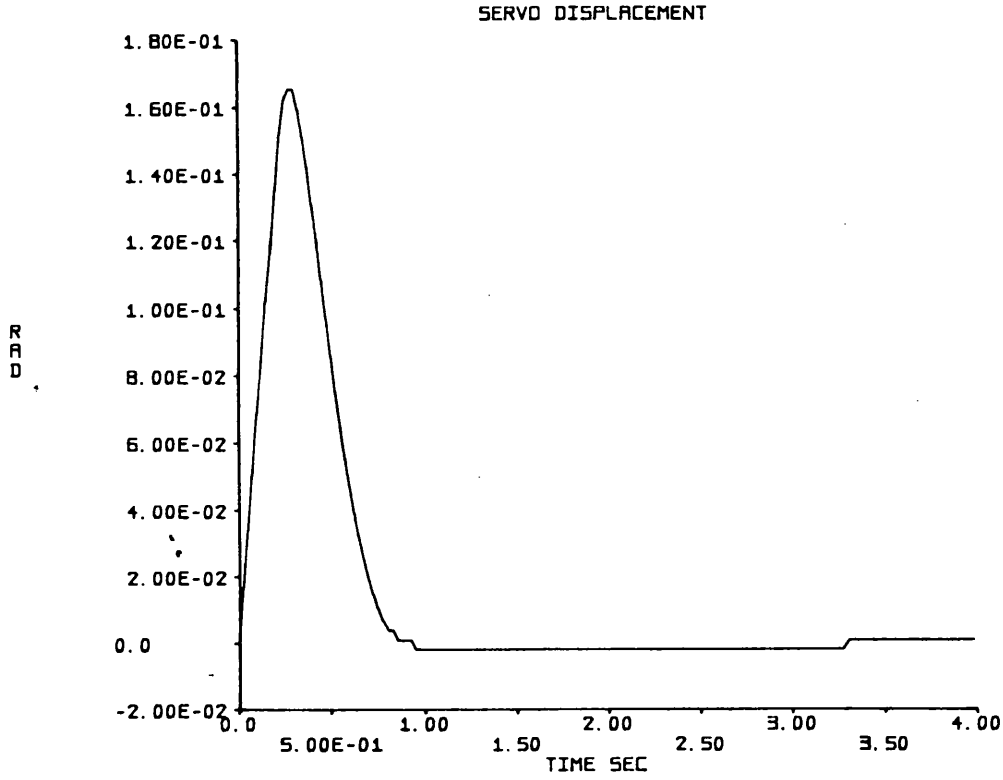


Fig. 62 Simulated aileron displacement, 40 m/s, gain = 0.4

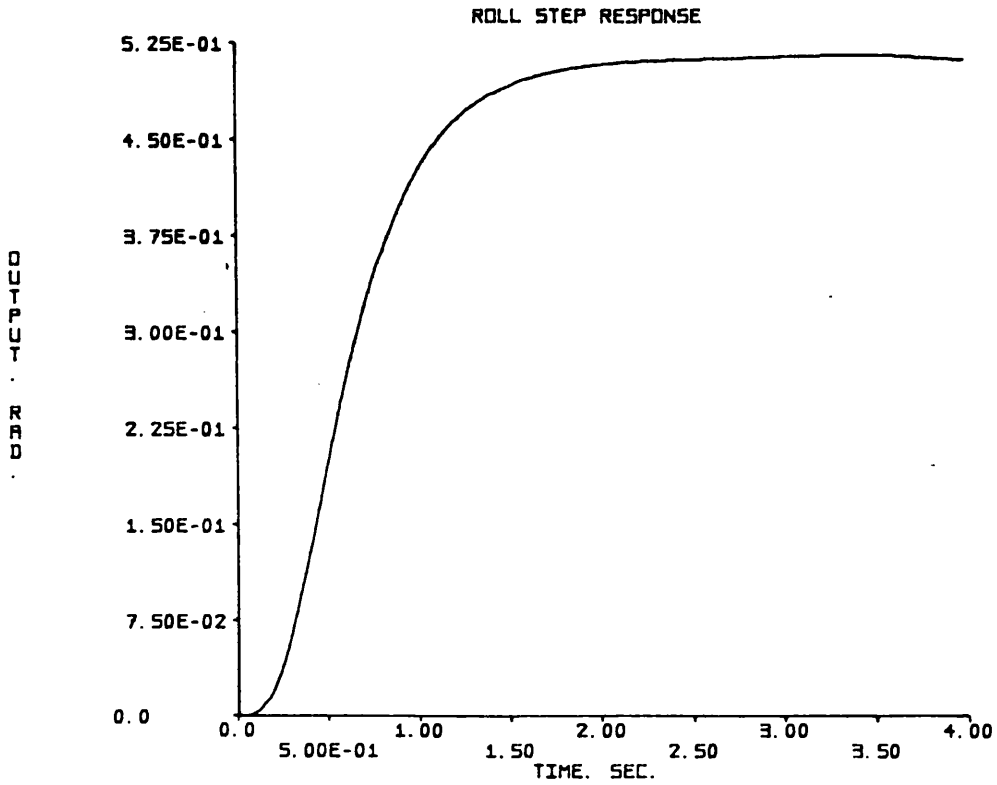


Fig. 63 Simulated roll attitude step response, 22 m/s, gain = 0.4

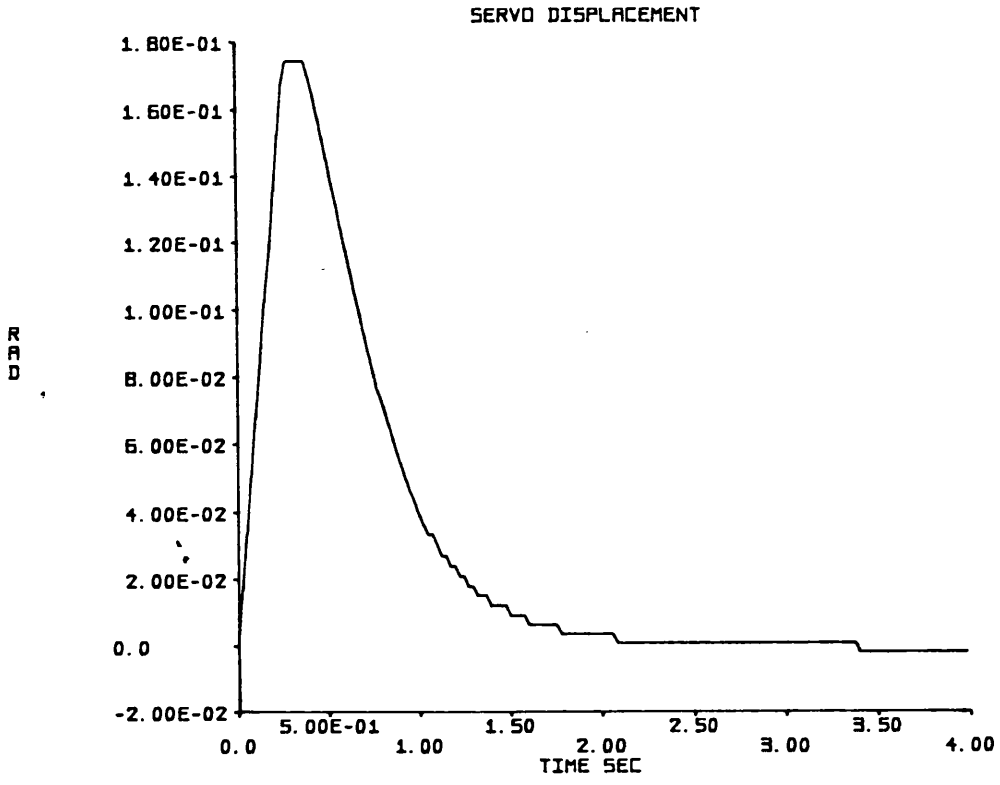


Fig. 64 Simulated aileron displacement, 22 m/s, gain = 0.4

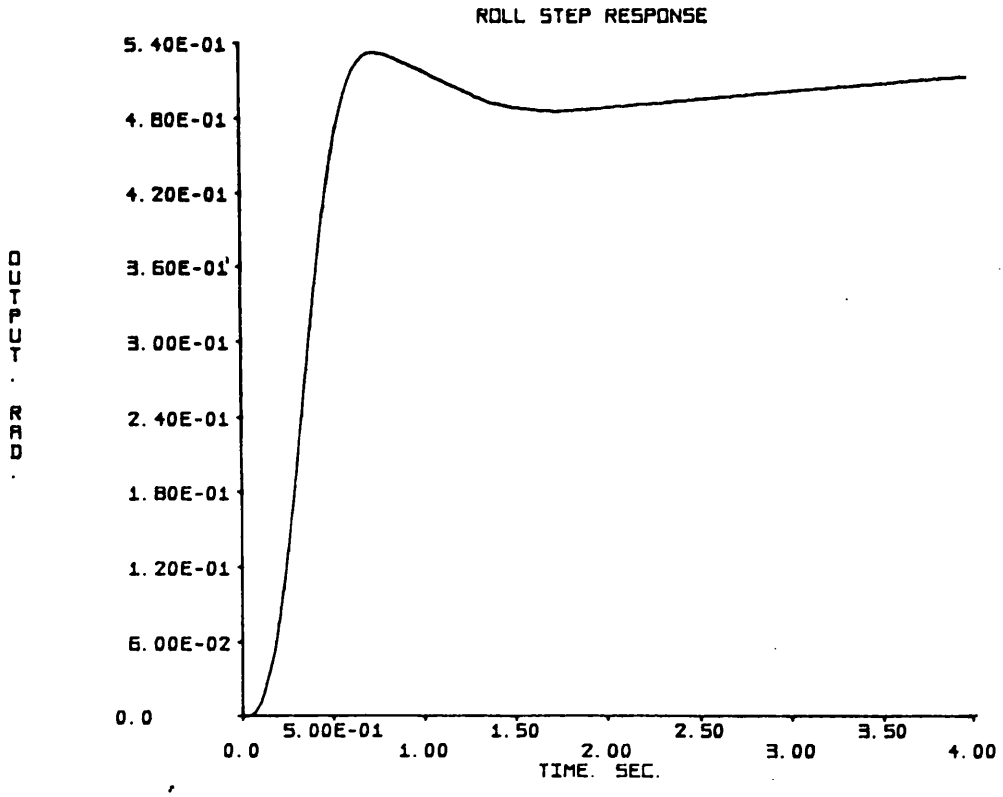


Fig. 65 Simulated roll attitude step response, 50 m/s, gain = 0.4

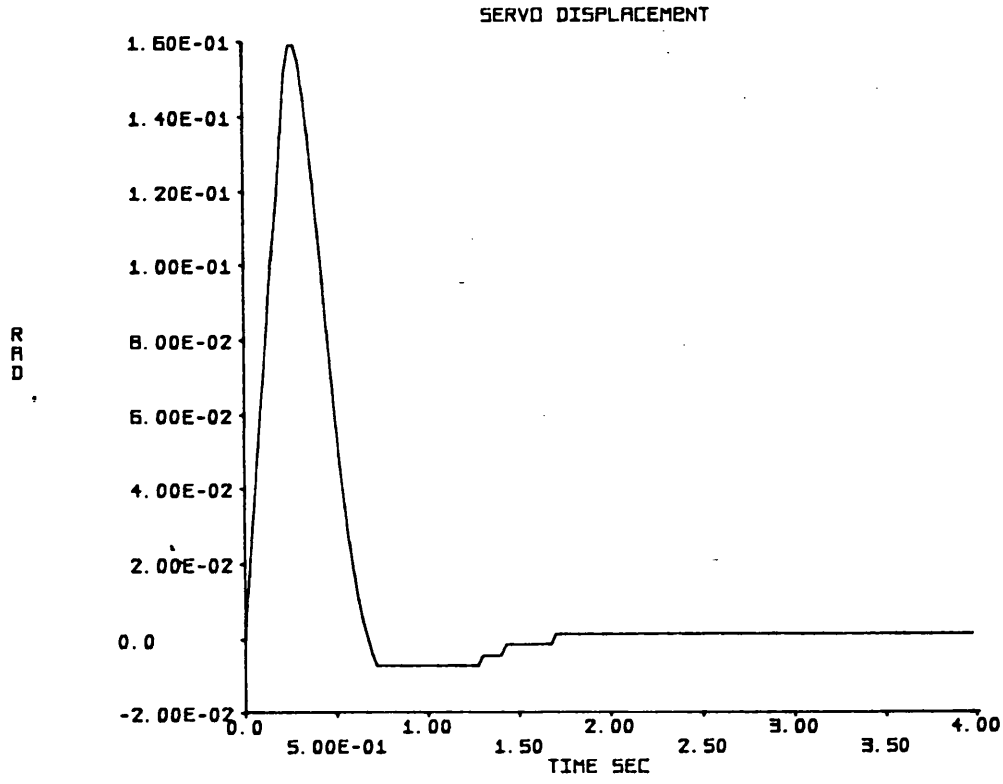


Fig. 66 Simulated aileron displacement, 50 m/s, gain =0.4

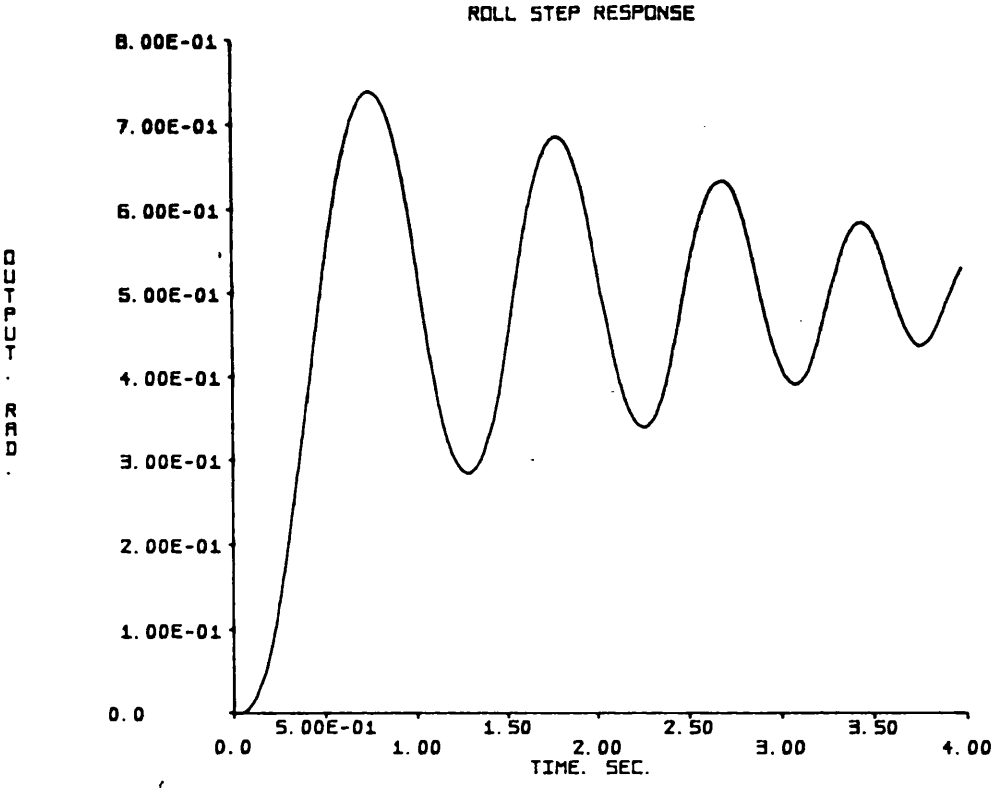


Fig. 67 Simulated roll attitude step response, 50 m/s, gain = 1.2

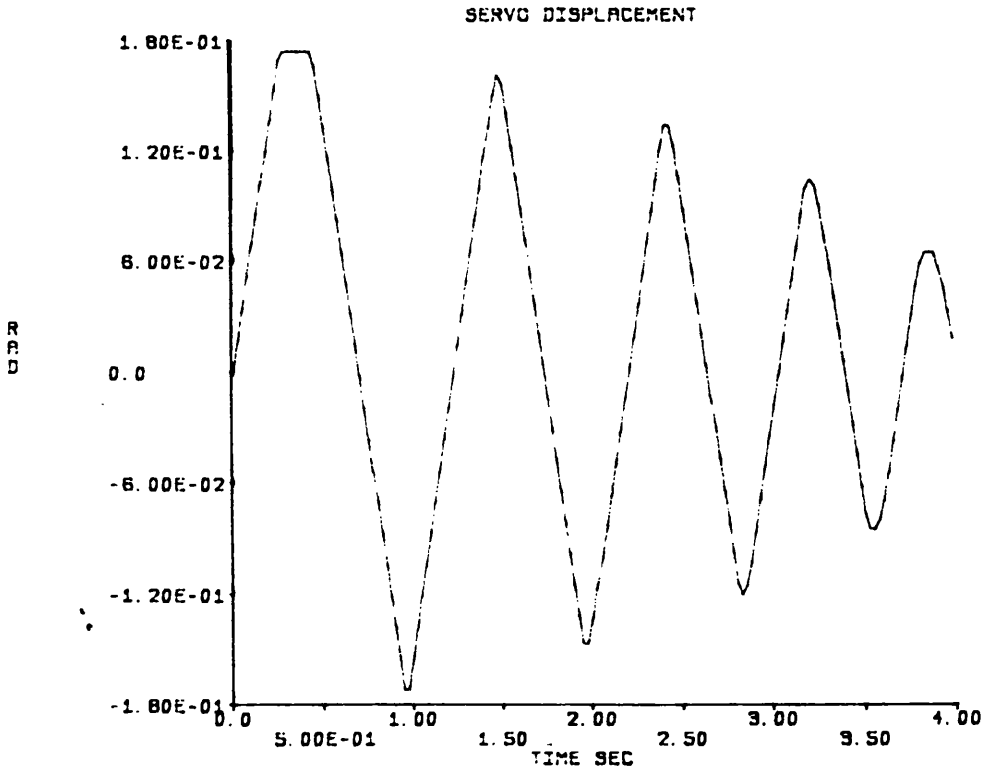


Fig. 68 Simulated servo-actuator displacement, 50 m/s, gain = 1.2

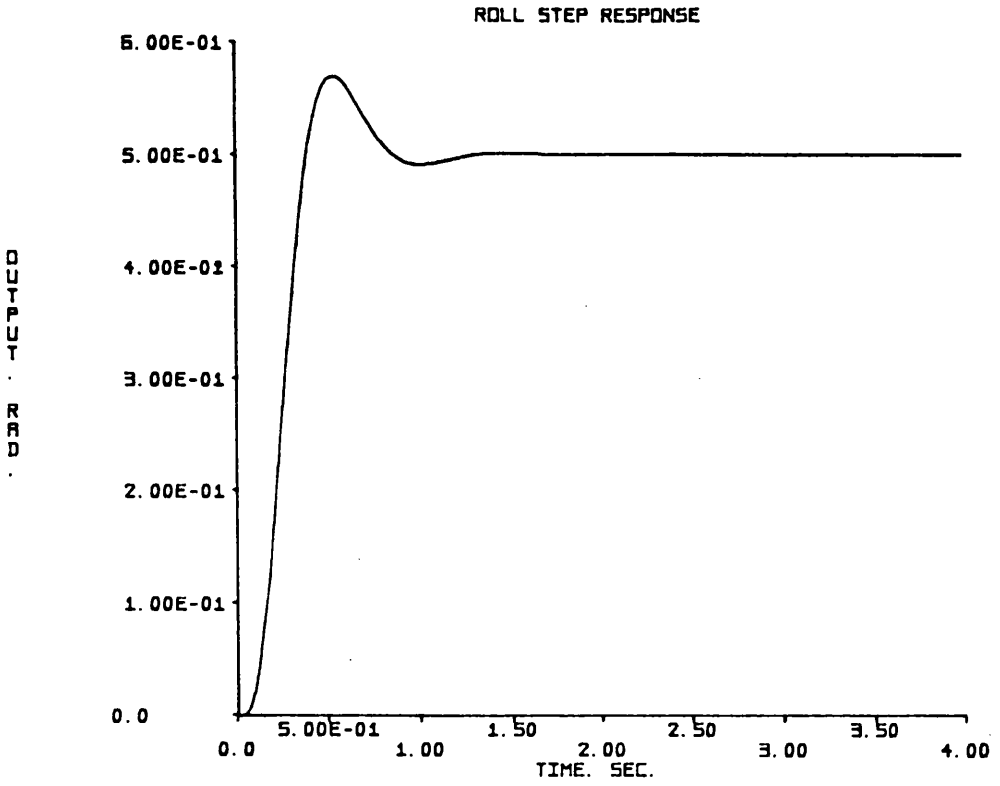


Fig. 69 Simulated roll attitude step response, 40 m/s, gain = 0.4, linearised actuator

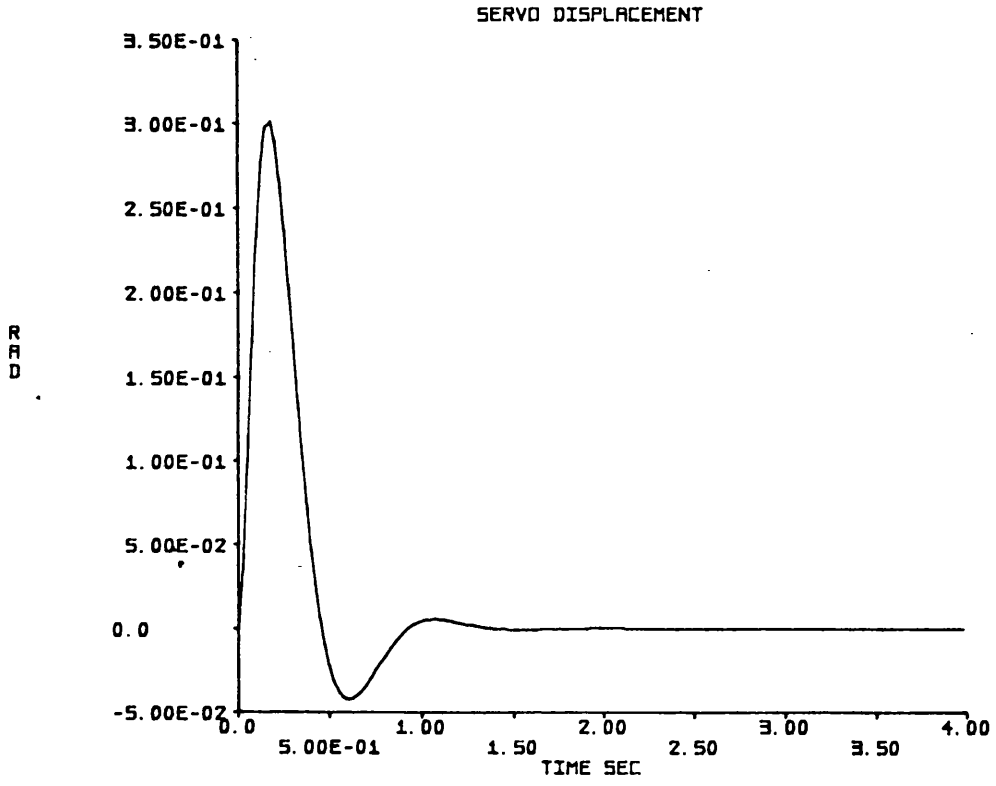


Fig. 70 Simulated aileron displacement, 40 m/s, gain = 0.4, linearised actuator

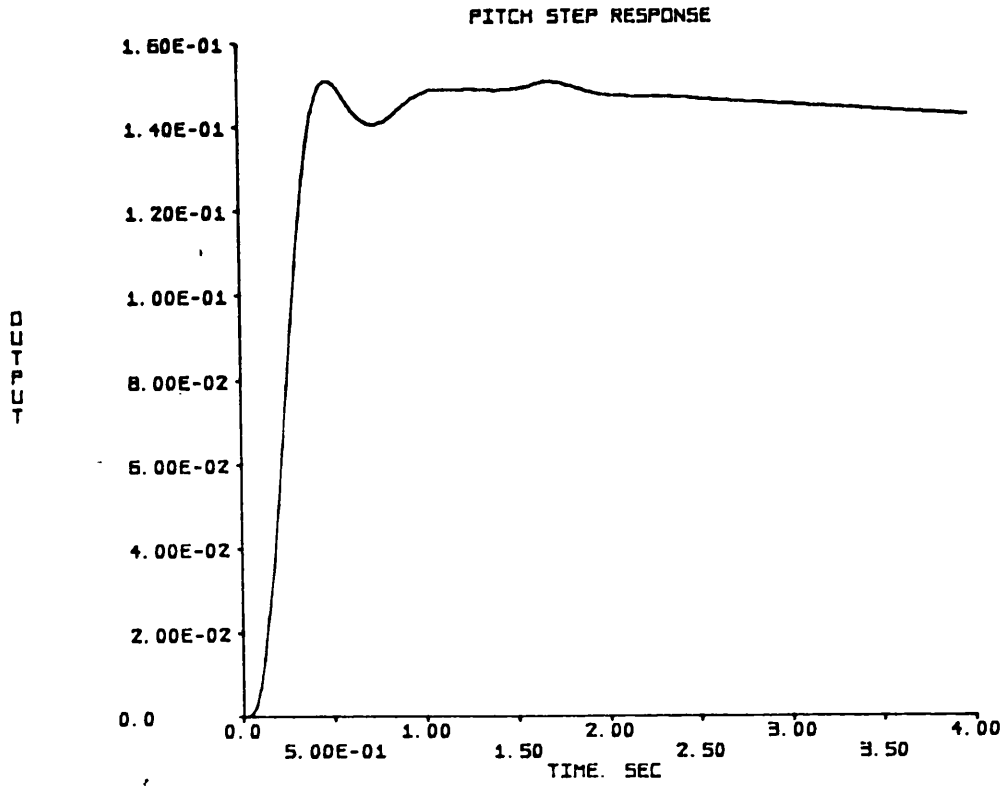


Fig. 71 Simulated pitch attitude step response, 40 m/s, gain = 0.86

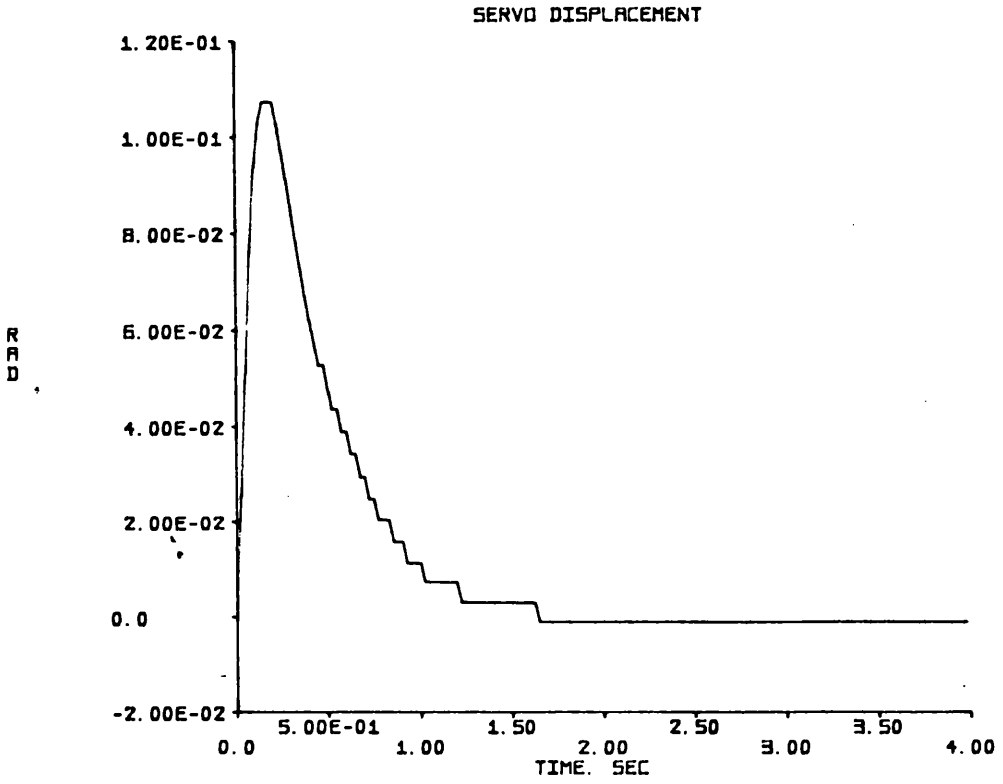


Fig. 72 Simulated elevator displacement, 40 m/s, gain = 0.86

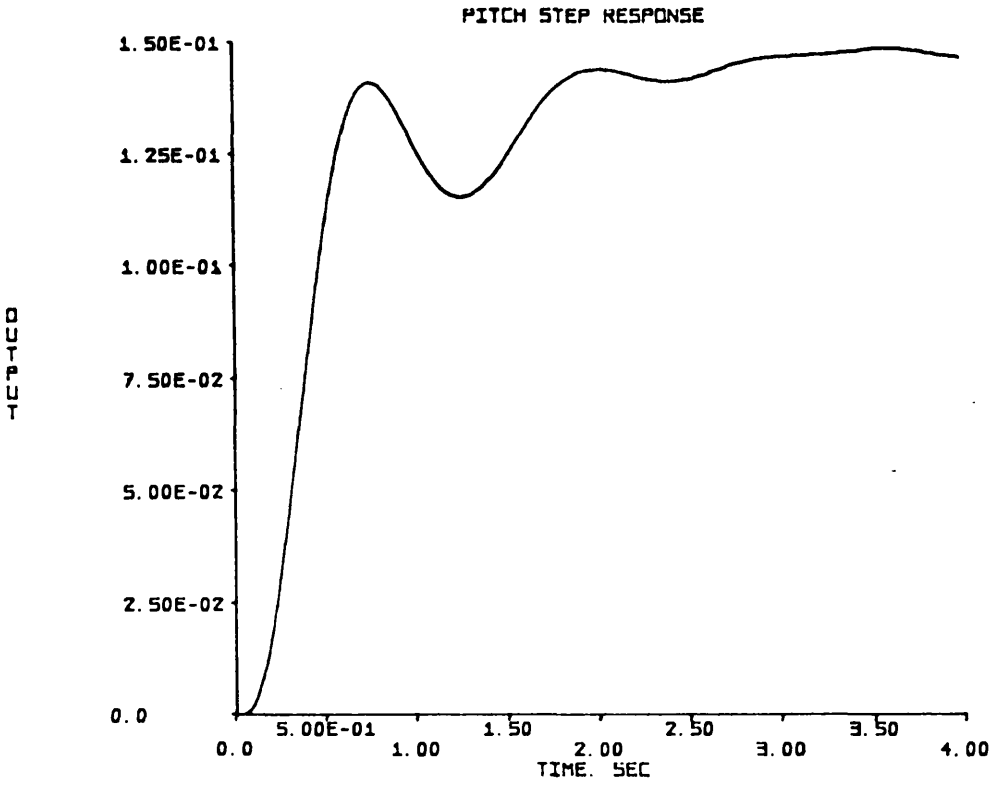


Fig. 73 Simulated pitch attitude step response, 22 m/s, gain = 0.86

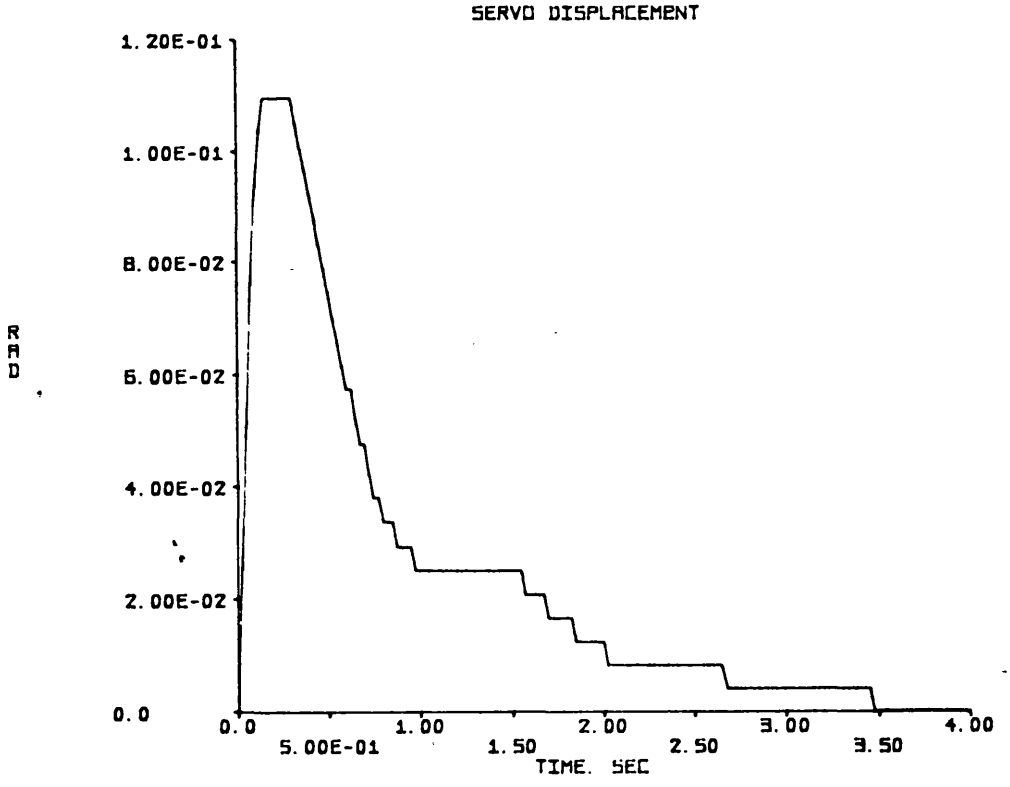


Fig. 74 Simulated elevator displacement, 22 m/s, gain = 0.86

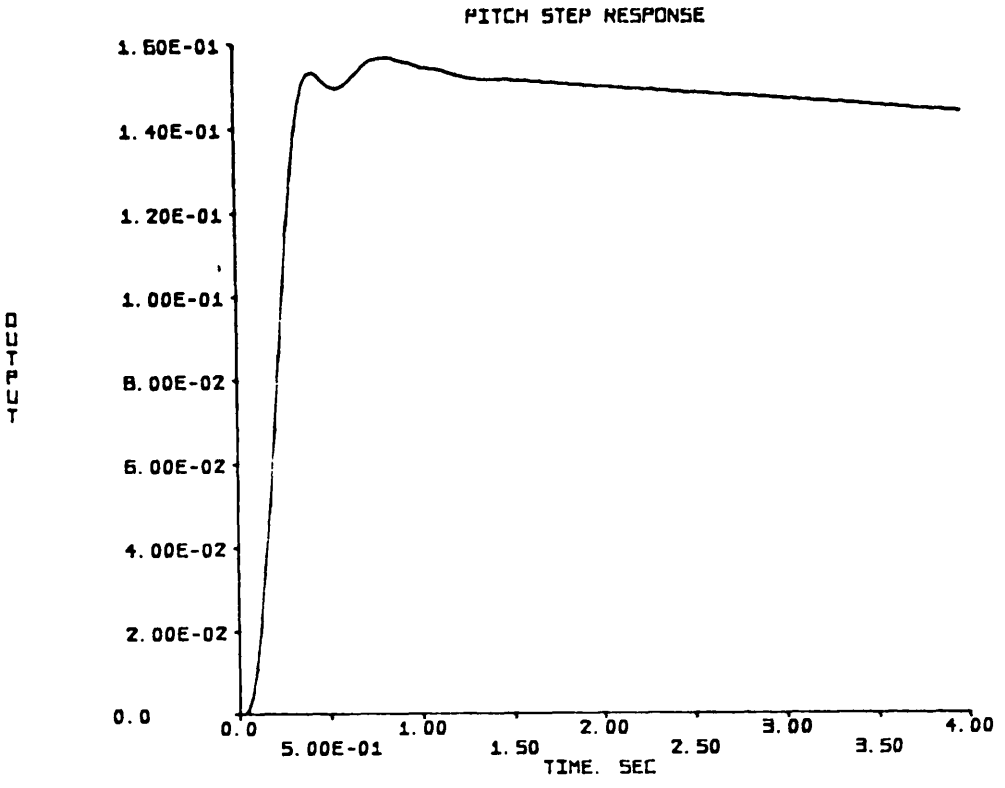


Fig. 75 Simulated pitch attitude step response, 50 m/s, gain = 0.86

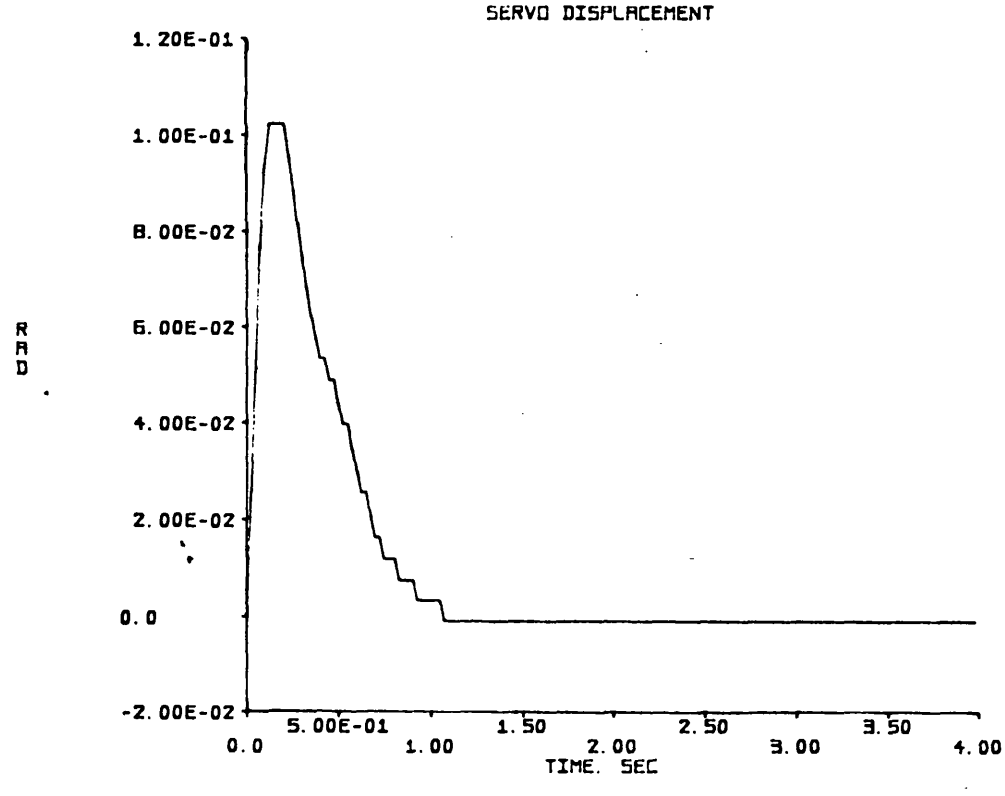


Fig. 76 Simulated elevator displacement, 50 m/s, gain = 0.86

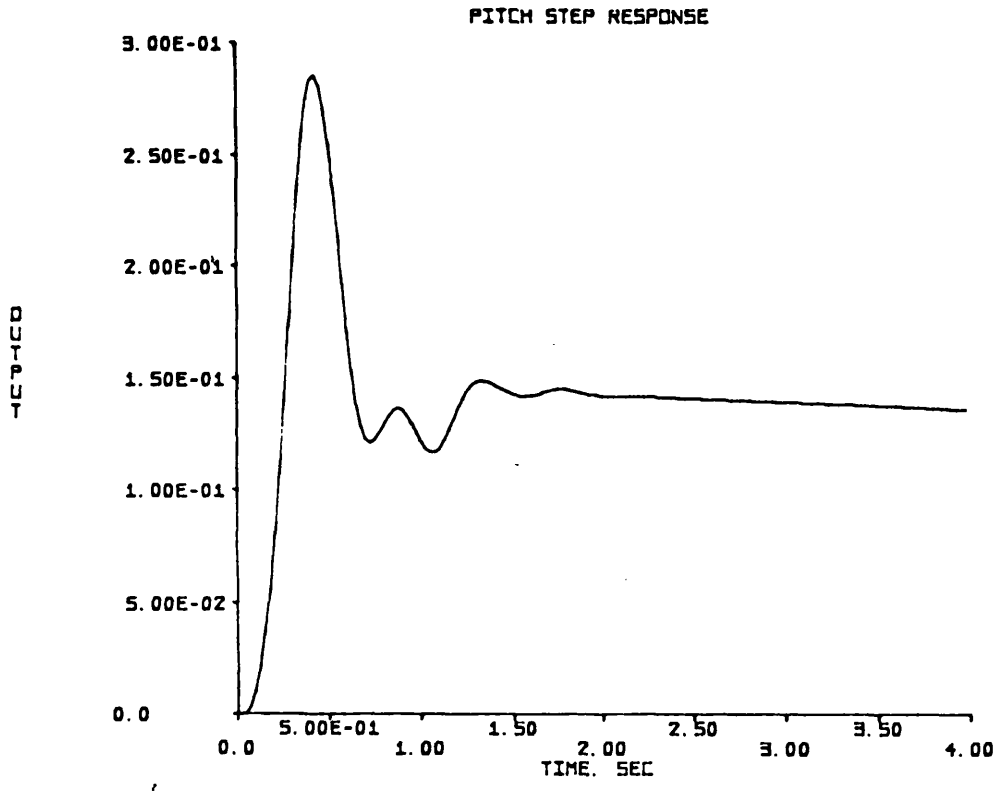
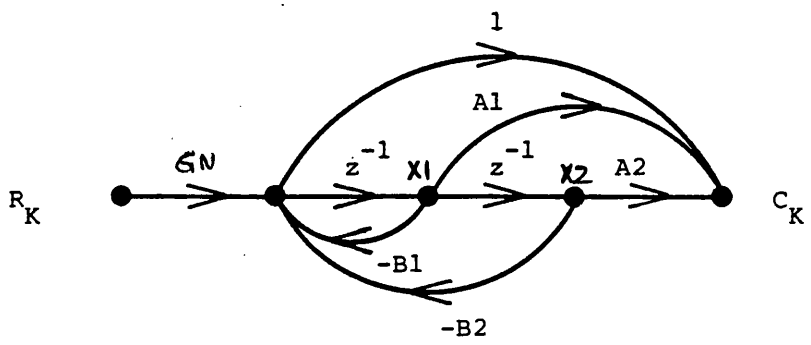


Fig. 77 Simulated pitch attitude step response, 50 m/s, gain = 2.58



The transfer function is $\frac{C(z)}{R(z)} = \frac{GN(1.0 + A1z^{-1} + A2z^{-2})}{(1.0 + B1z^{-1} + B2z^{-2})}$

FIG. 78 The state diagram for the pitch compensator.

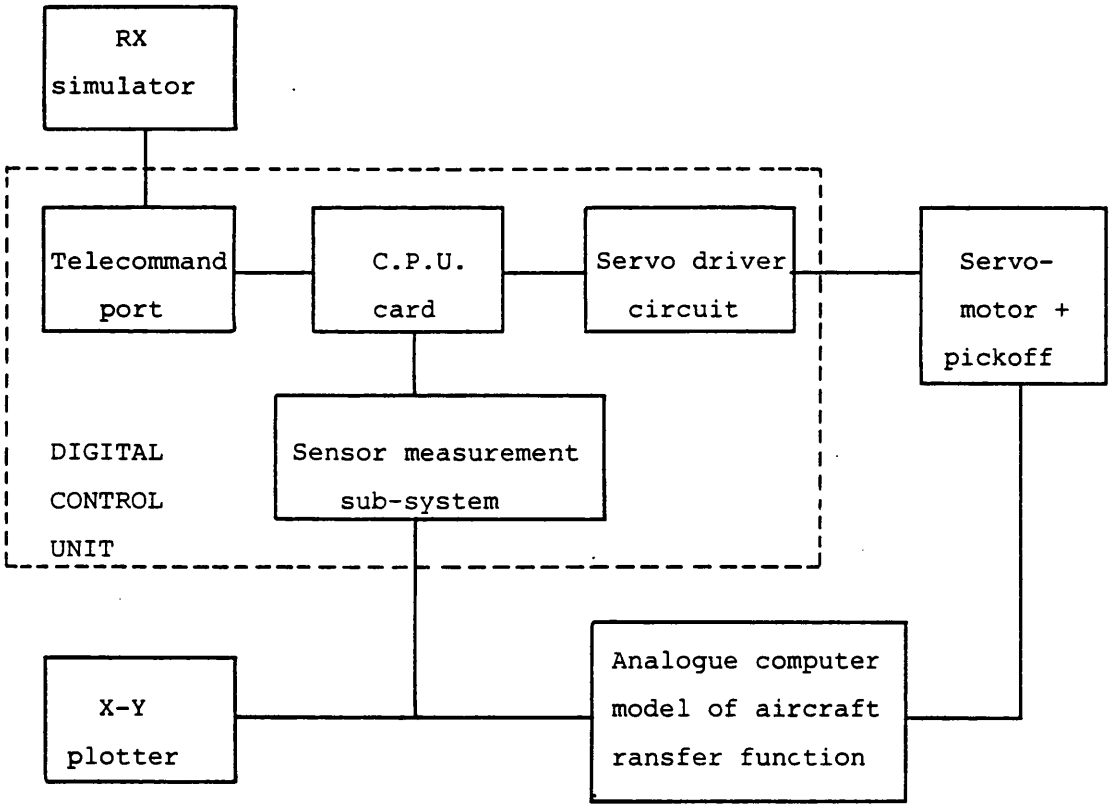


FIG. 79 Block diagram of the hybrid simulation.

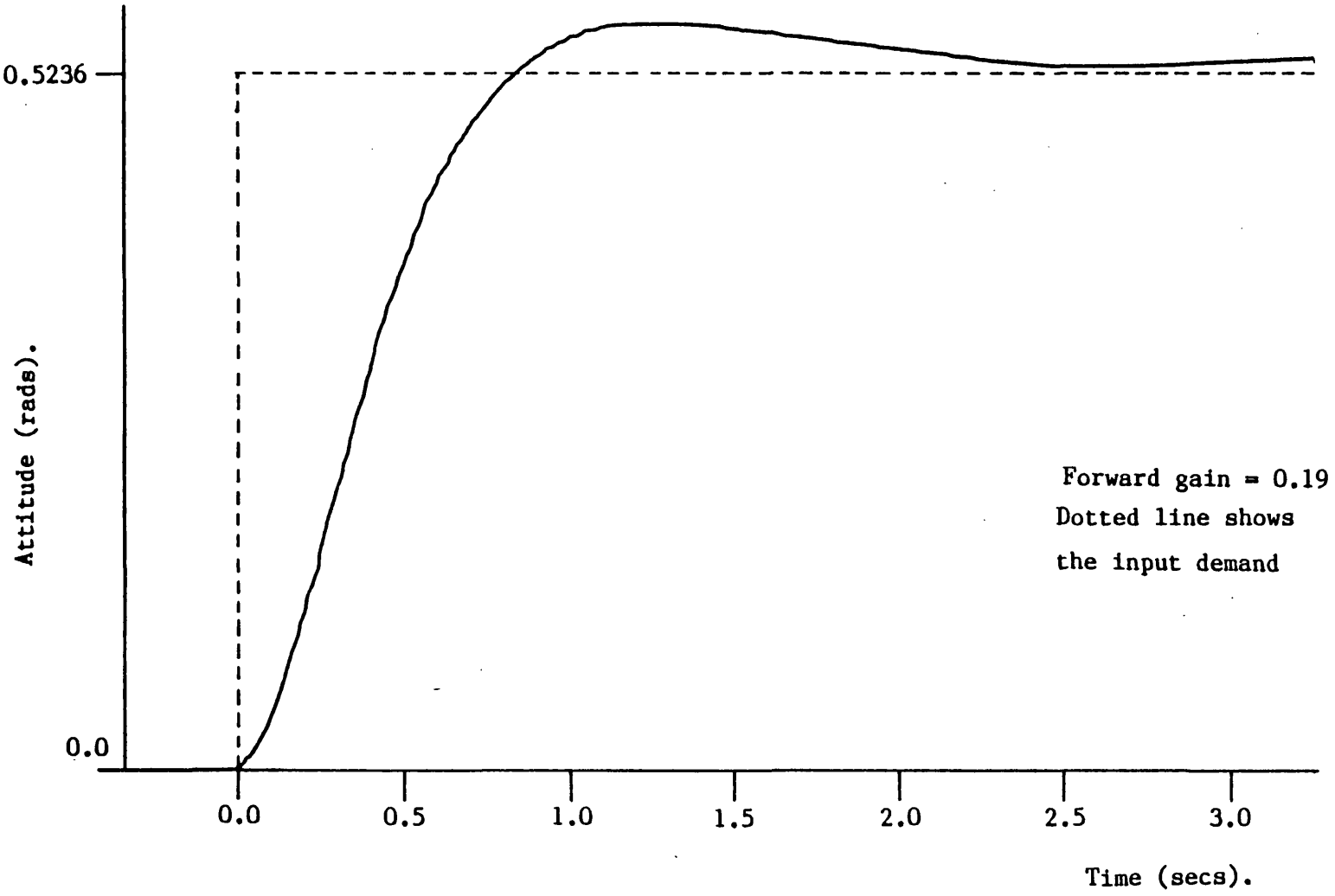


Fig. 80a Hybrid simulation result for the Mkl Stabileye roll autopilot, 30 m/s.

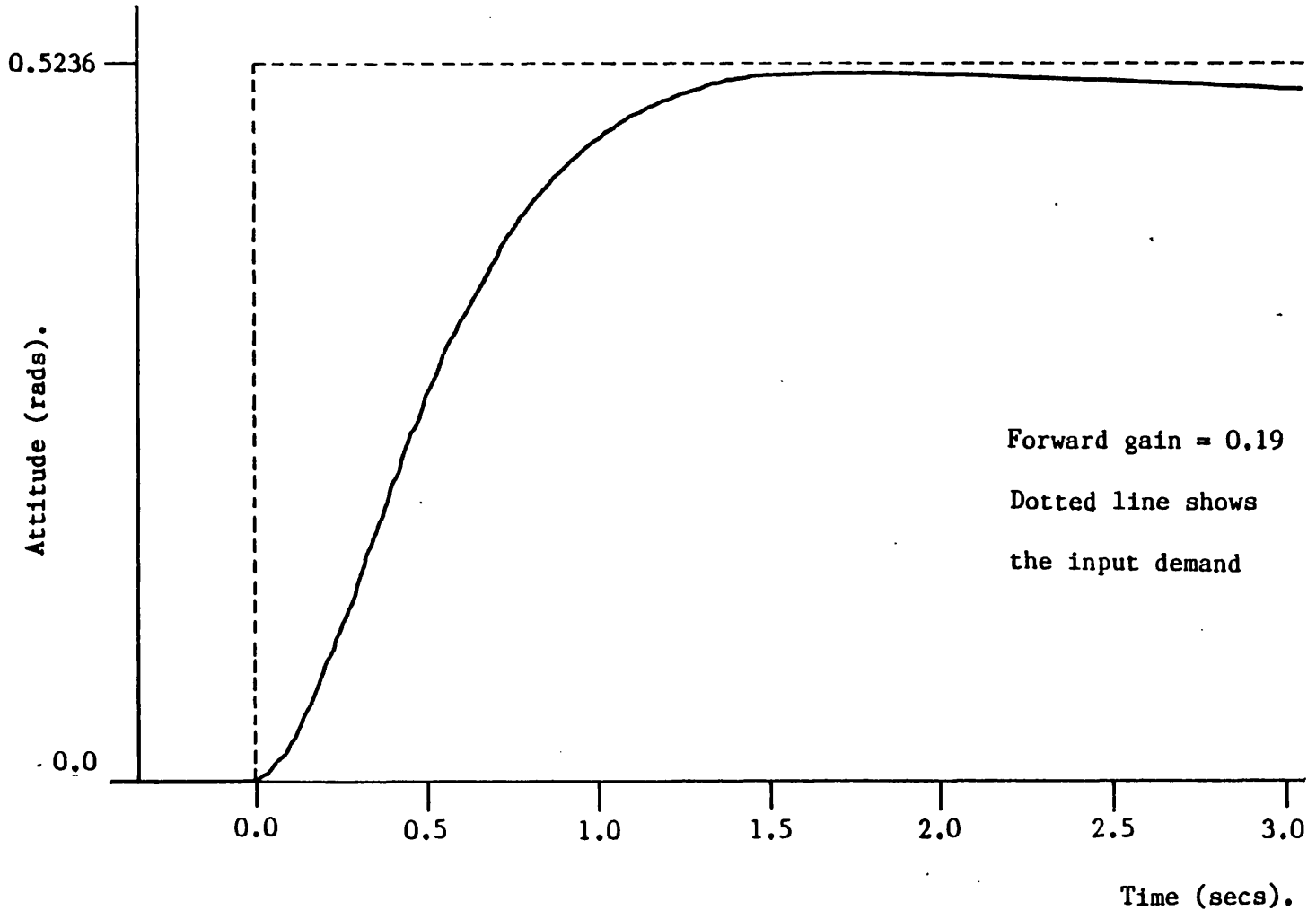


Fig. 80b Hybrid simulation result for the Mk1 Stabileye roll autopilot, 22 m/s.

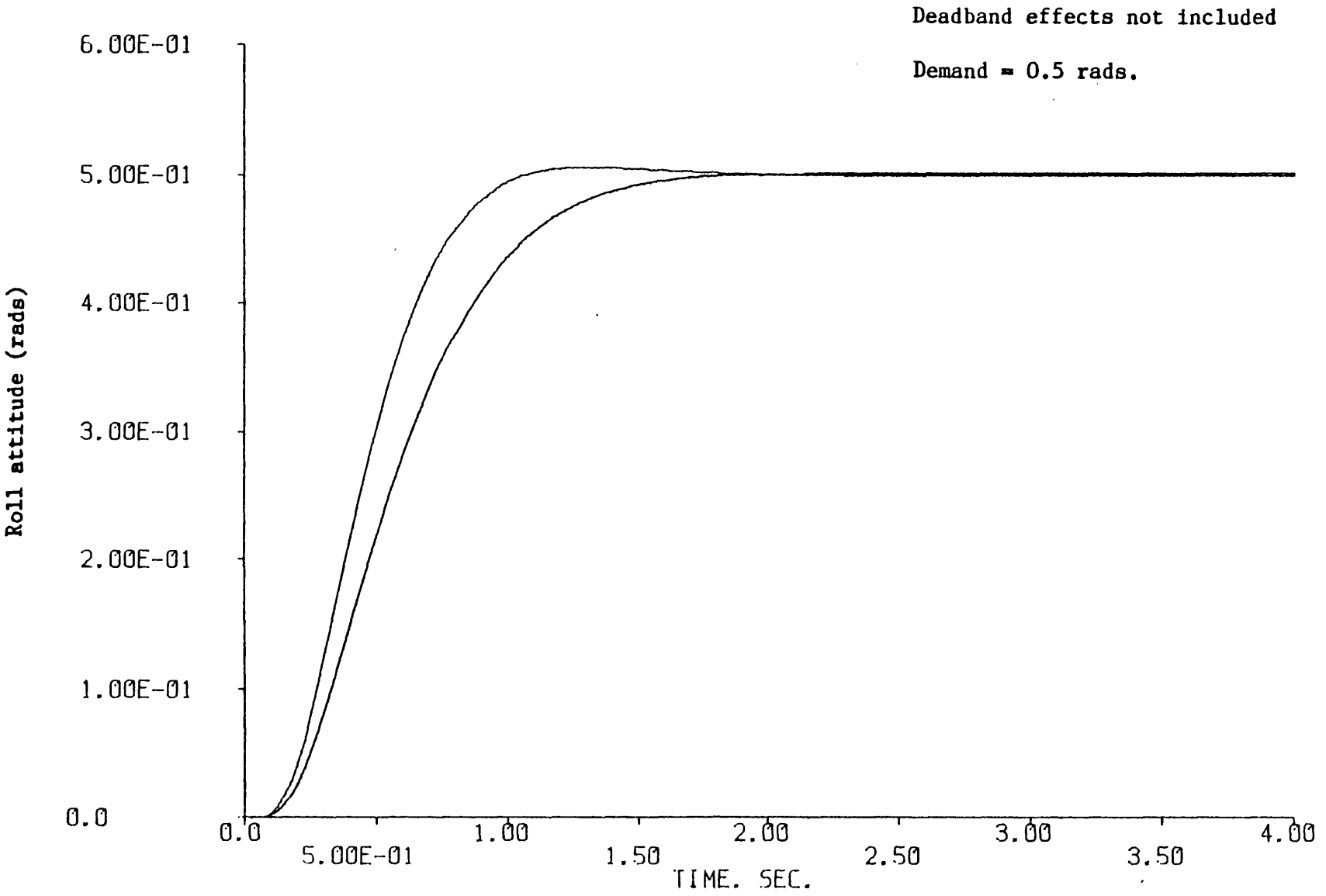


Fig. 80c Digital simulation of the Mkl Stabileye roll autopilot, 22 m/s and 30 m/s.

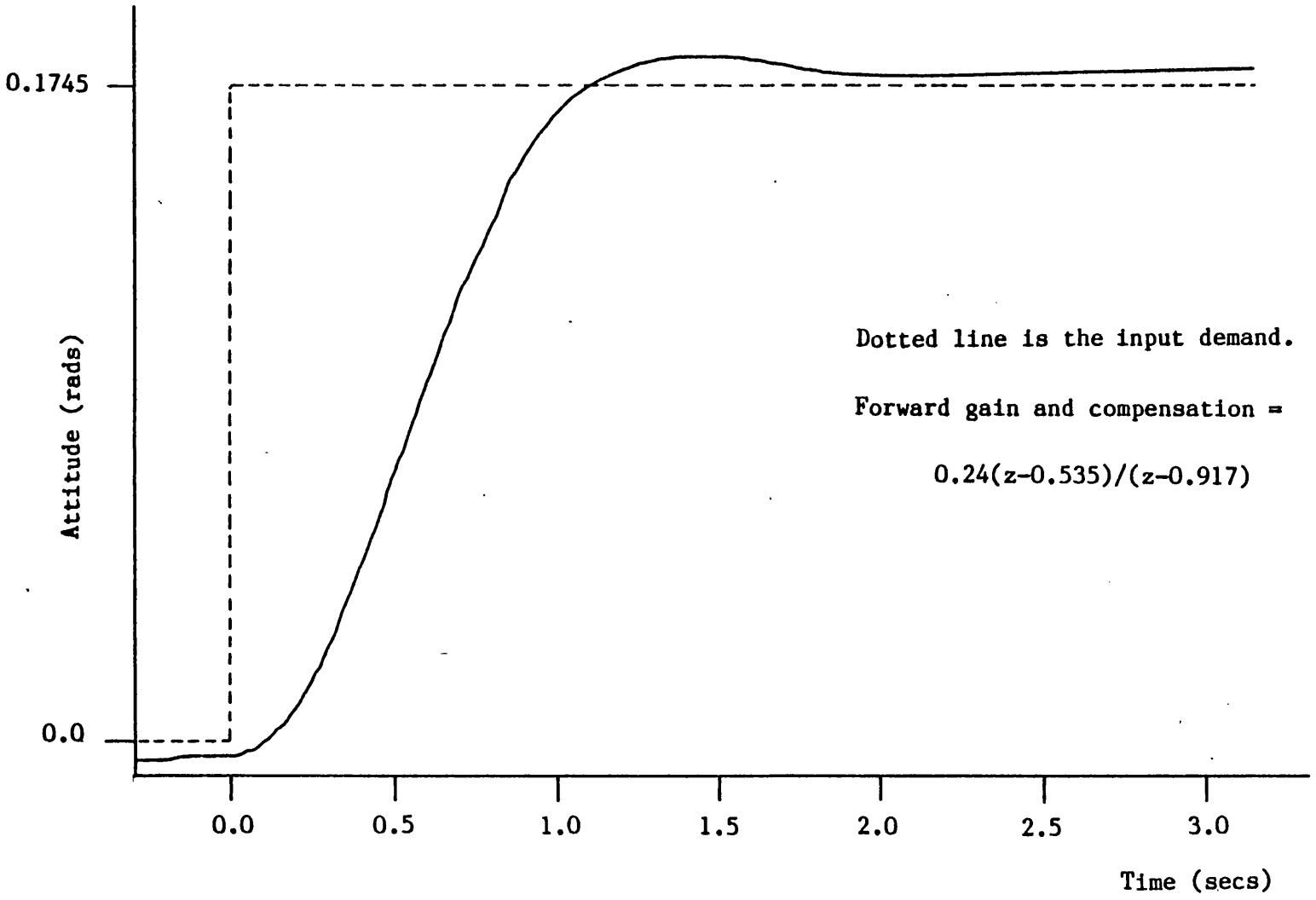


Fig. 81a Hybrid simulation result for the Mk1 Stabileye pitch autopilot, 30 m/s.

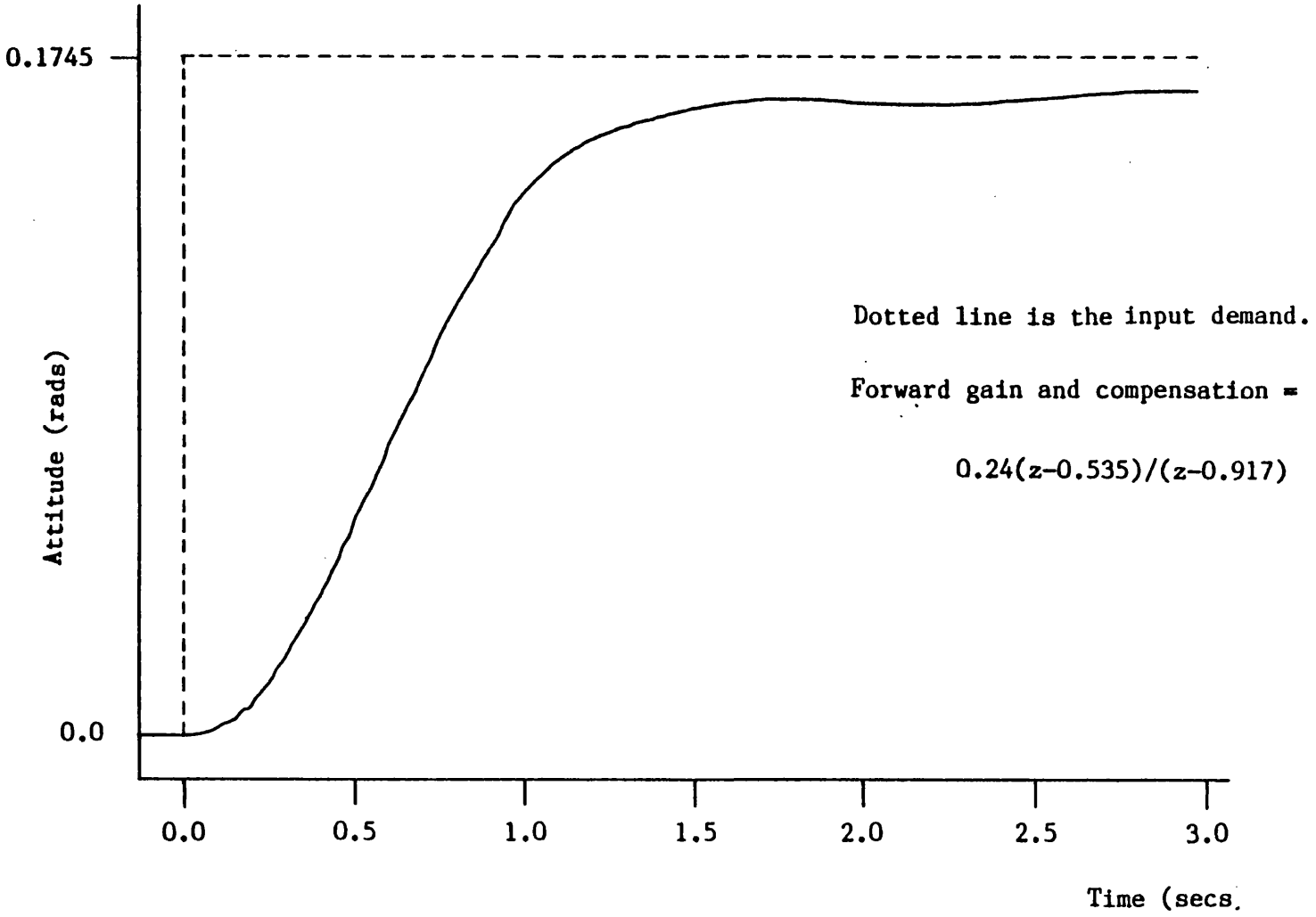


Fig. 81b Hybrid simulation result for the Mk1 Stabileye pitch autopilot, 22 m/s.

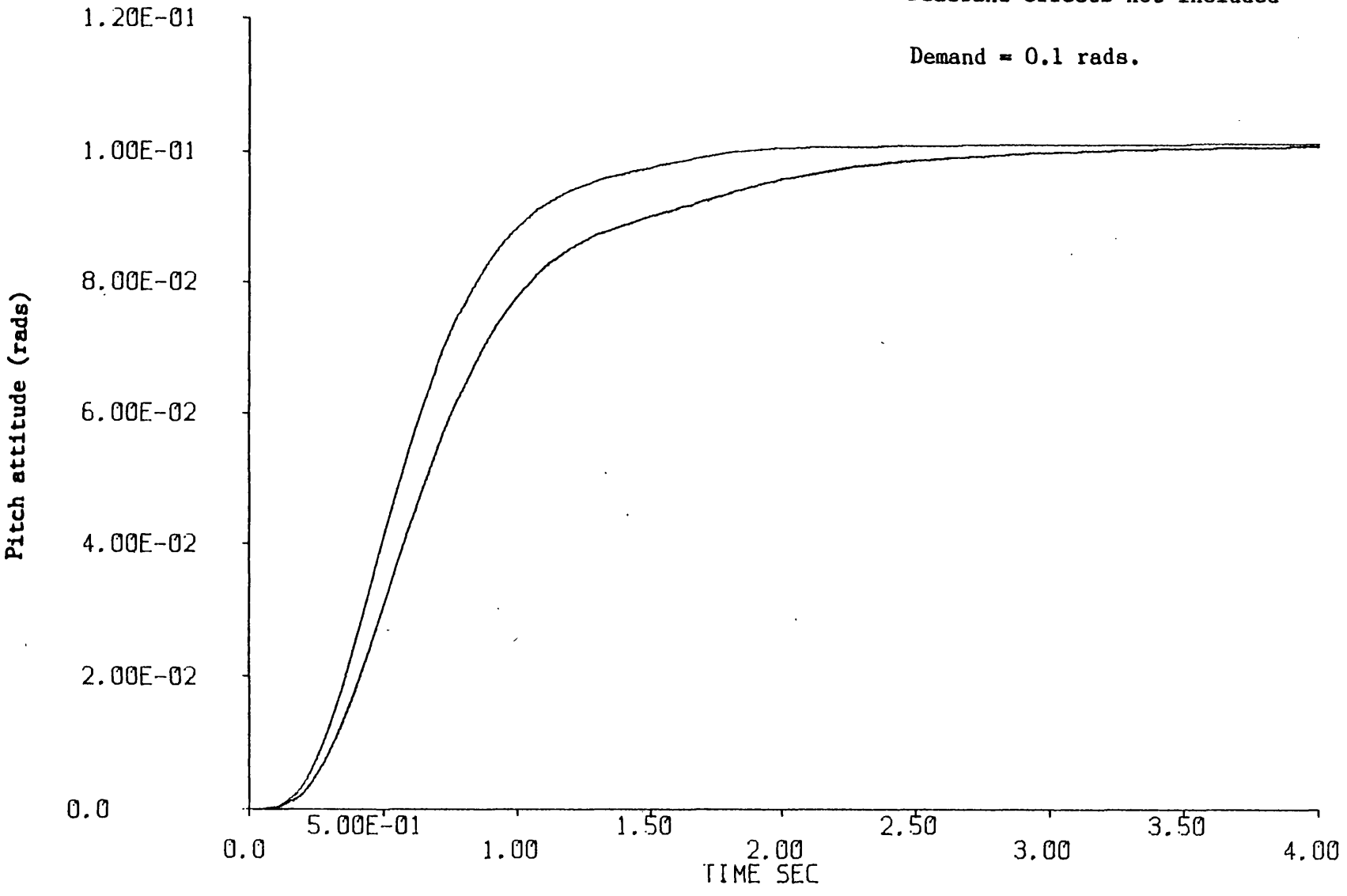


Fig. 81c Digital simulation of the Mk1 Stabileye pitch autopilot, 22 m/s and 30 m/s.

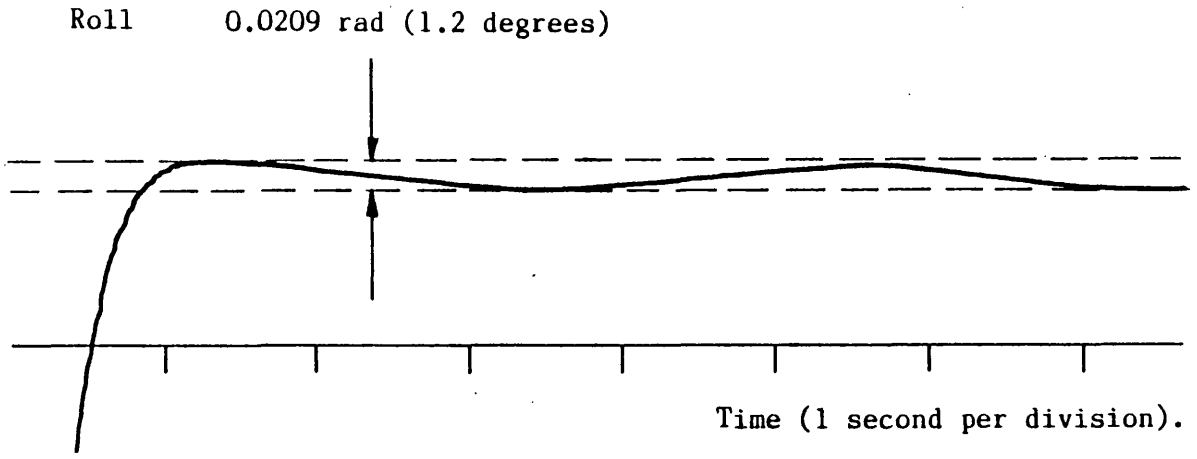


Fig. 82a Hybrid simulation of the Mkl Stabileye roll autopilot at steady state.

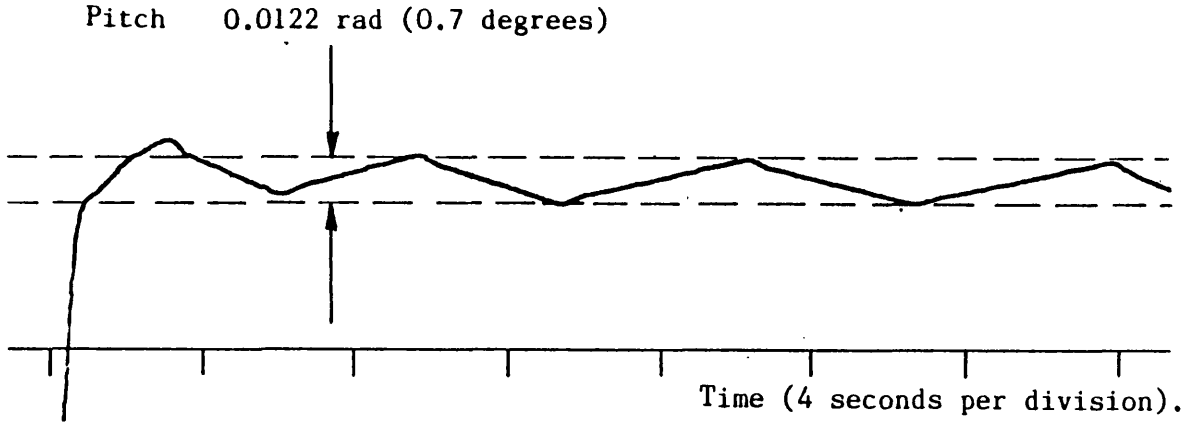


Fig. 82b Hybrid simulation of the Mkl Stabileye pitch autopilot at steady state.

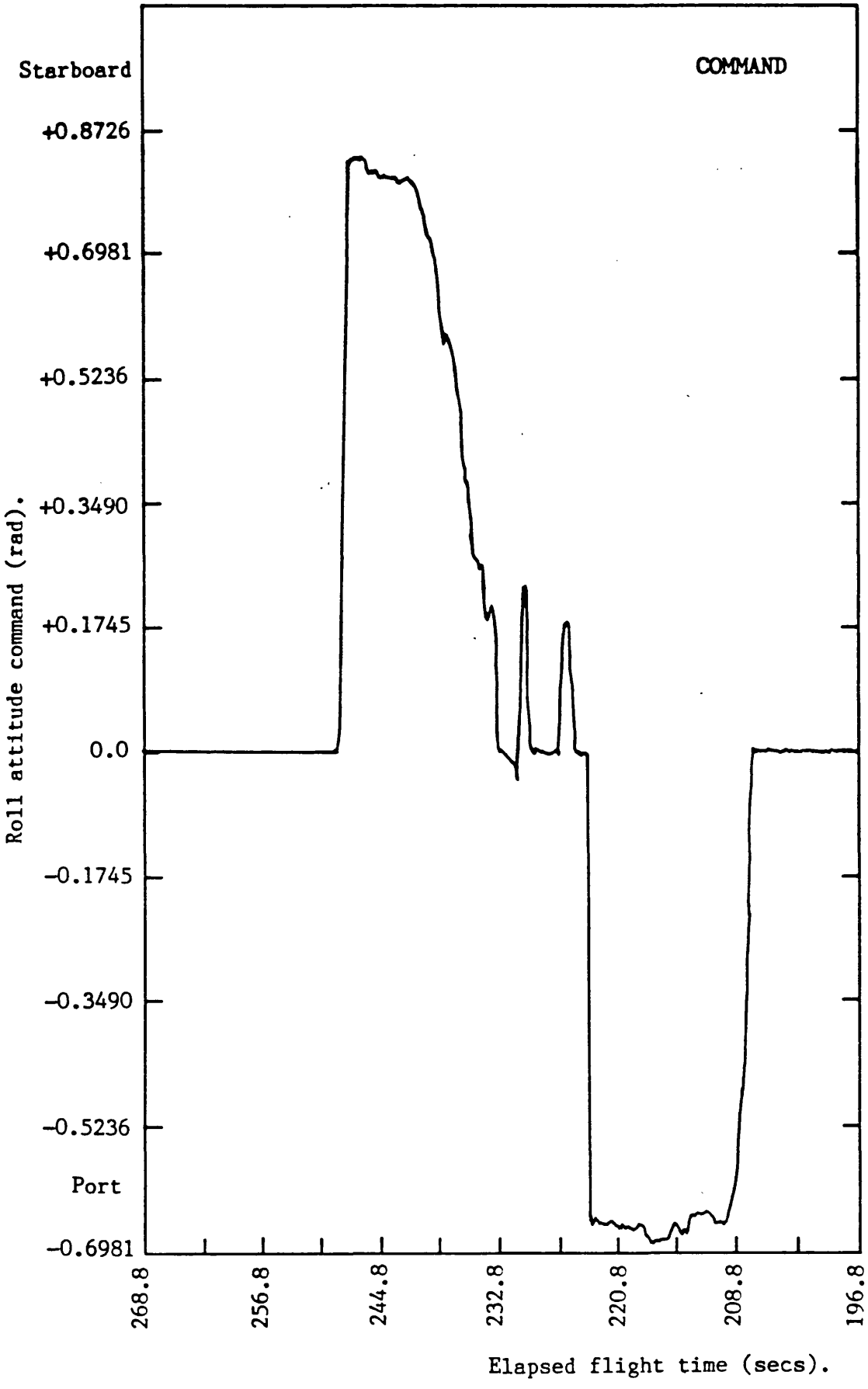
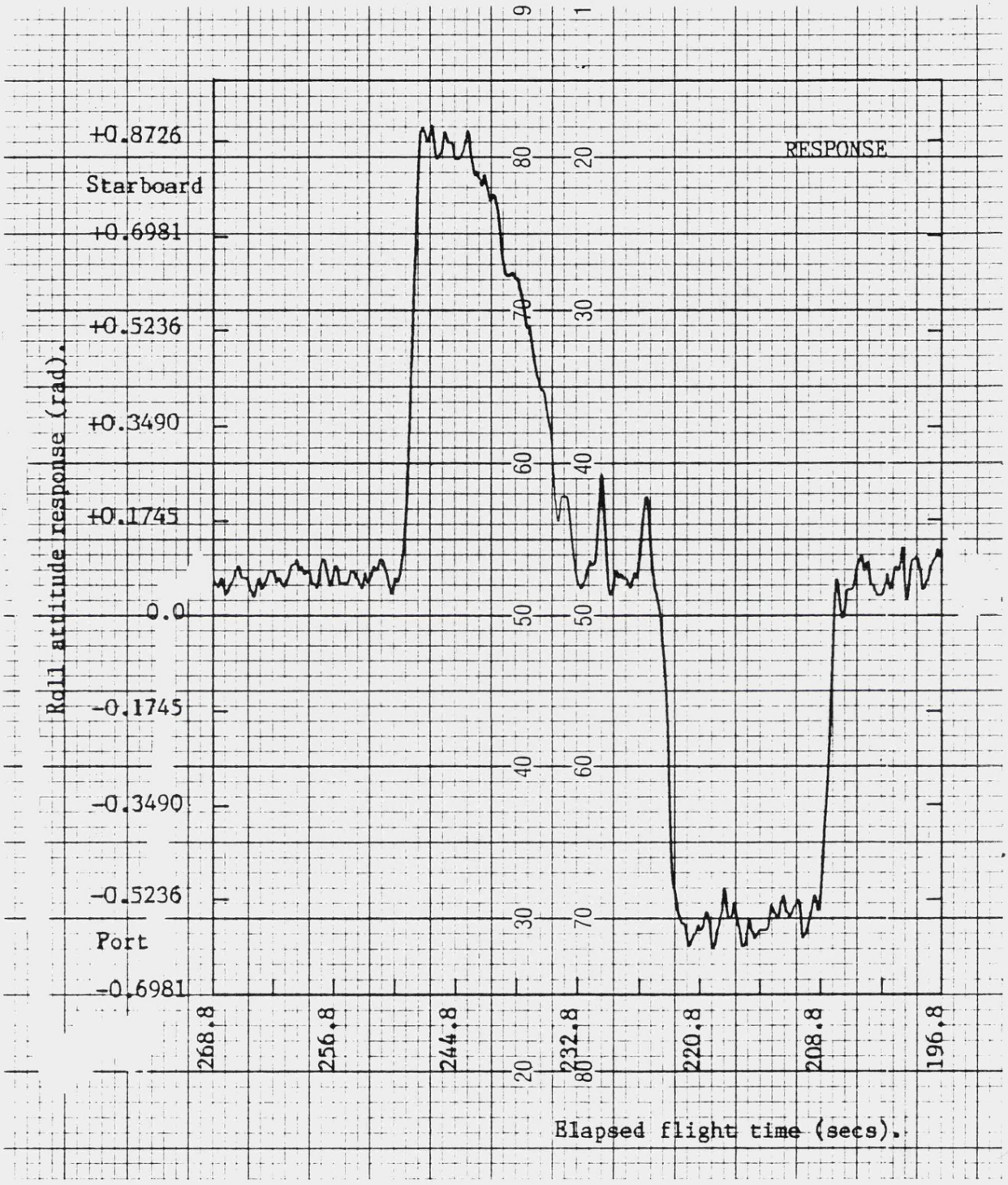


Fig. 83a Flight results: Roll command.



Window corresponds to Fig. 83a

Fig. 83b Flight results: Roll response.

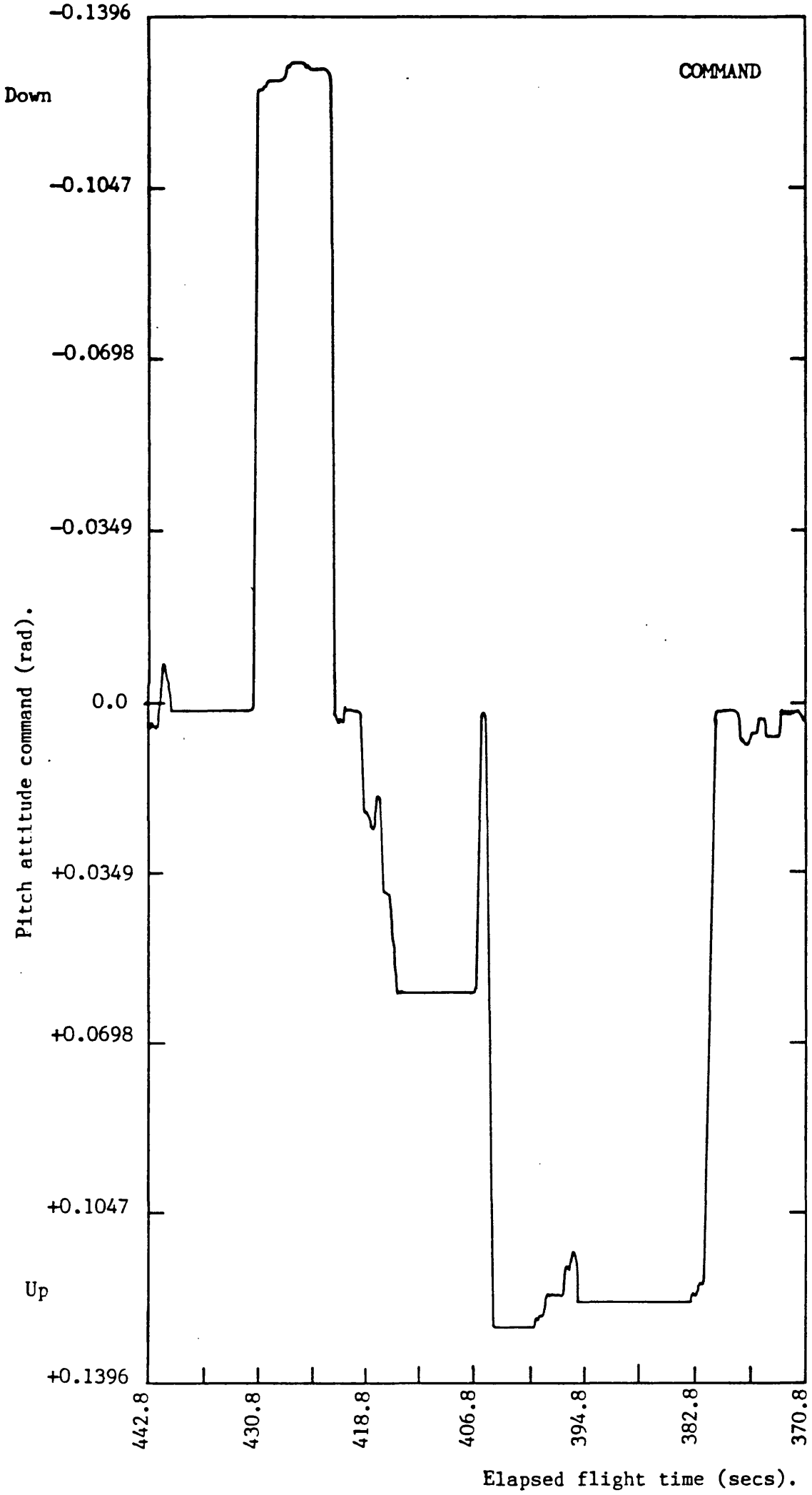
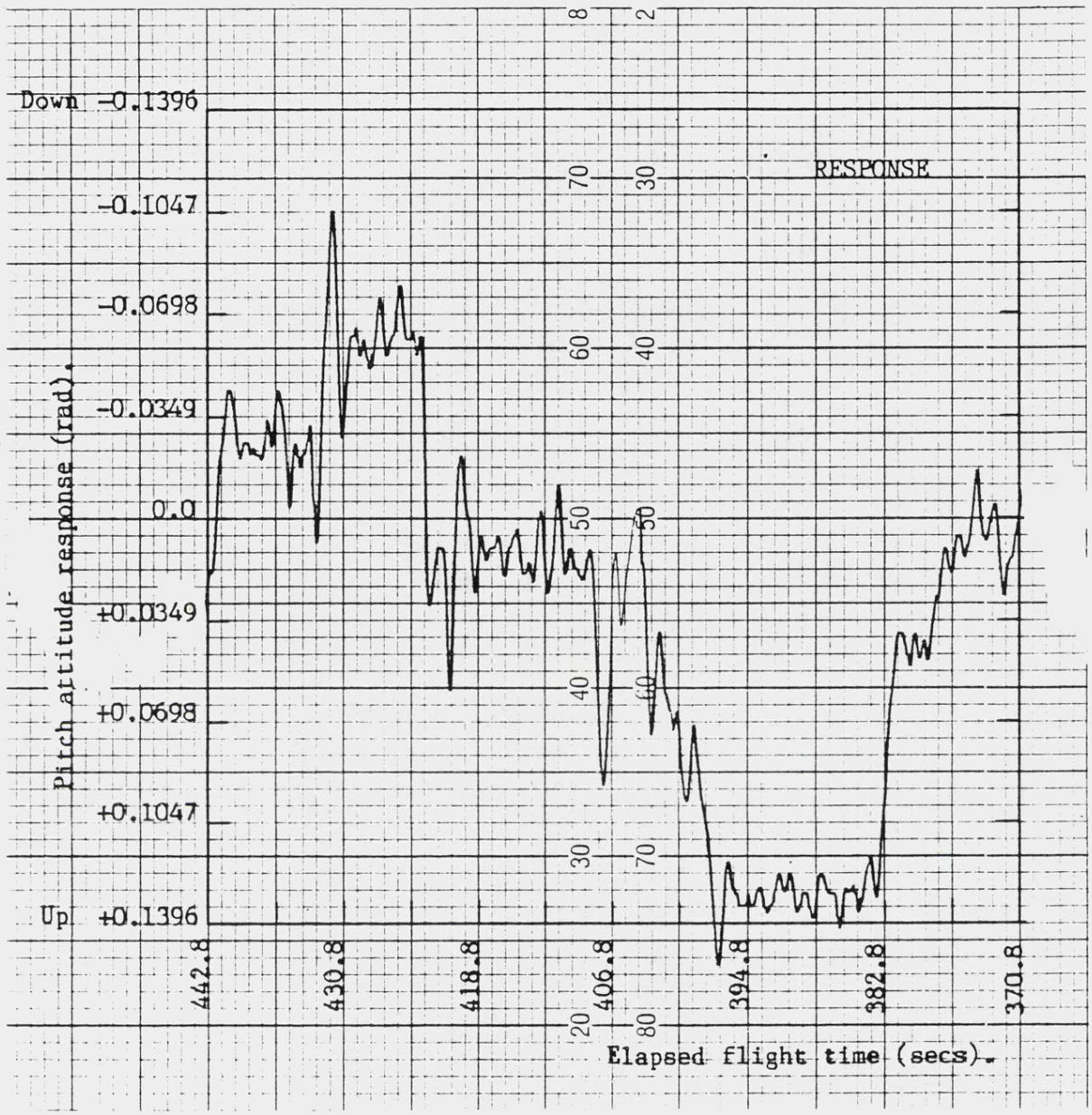


Fig. 84a Flight results: Pitch command.



Window corresponds to Fig. 84a

Fig. 84b Flight results: Pitch response.

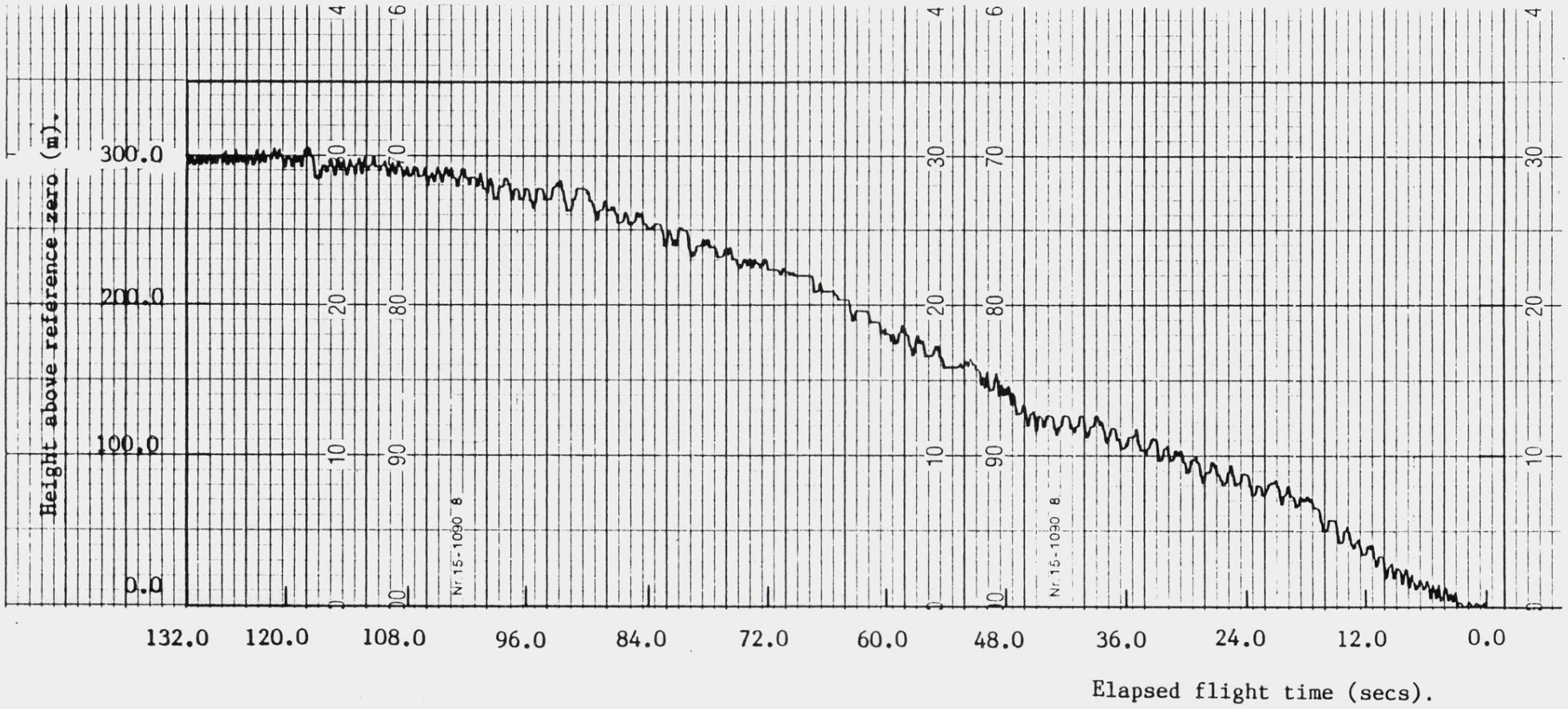


Fig. 85 Flight results: Automatic climb to height. Demanded height = 300.0m.

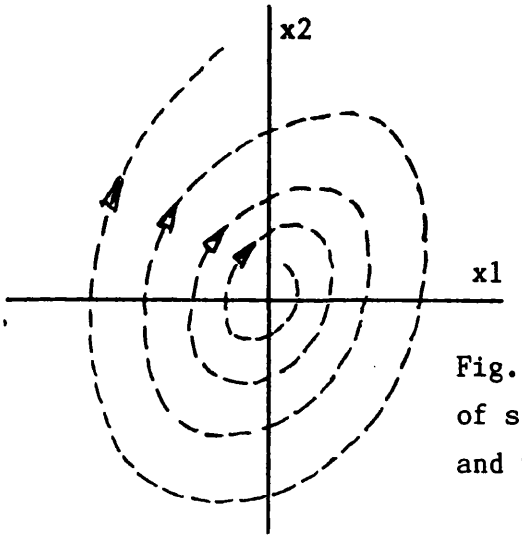


Fig. 86a Phase plane of system in eqns. 9.1 and 9.2, with $\psi = \alpha$.

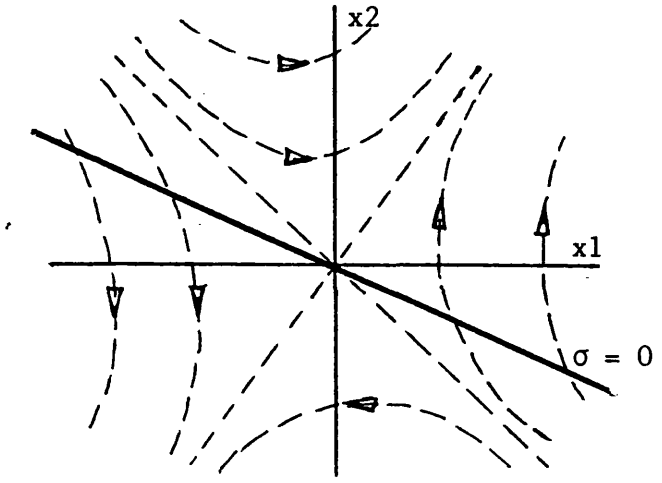


Fig. 86b Phase plane of system in eqns. 9.1 and 9.2 with $\psi = -\alpha$.

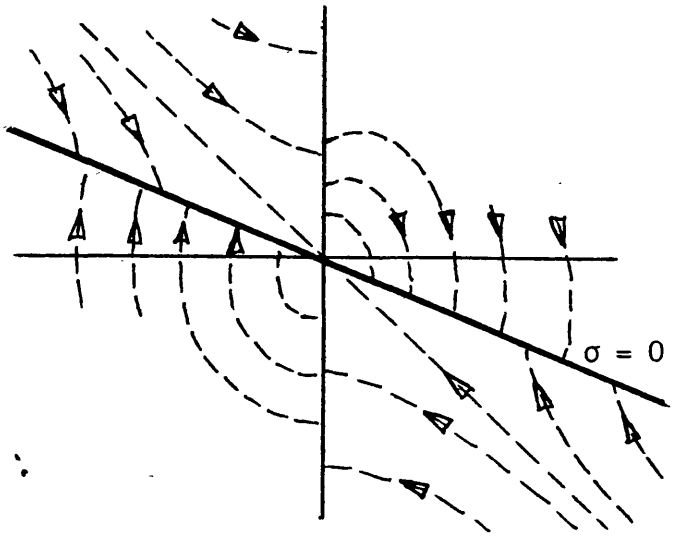


Fig. 86c Phase plane of variable structure system based on eqns. 9.1, 9.2, 9.3, 9.4.

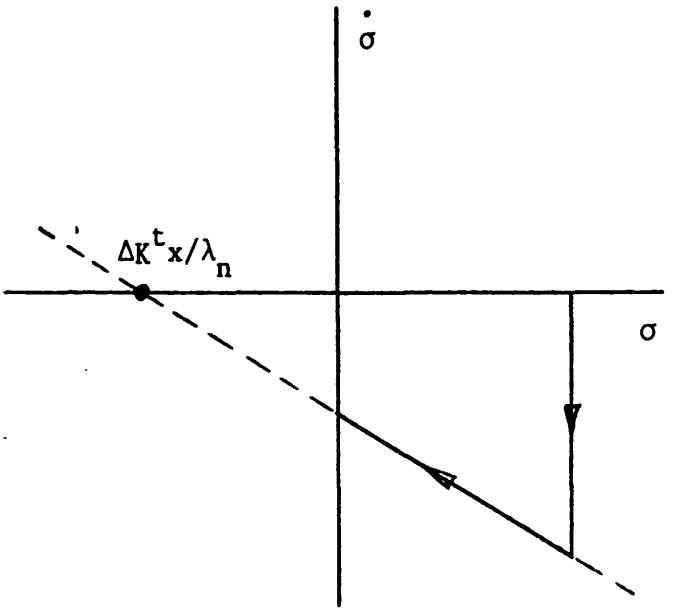


Fig. 87a Straight line approximation of full order σ range space trajectory, λ_n stable.

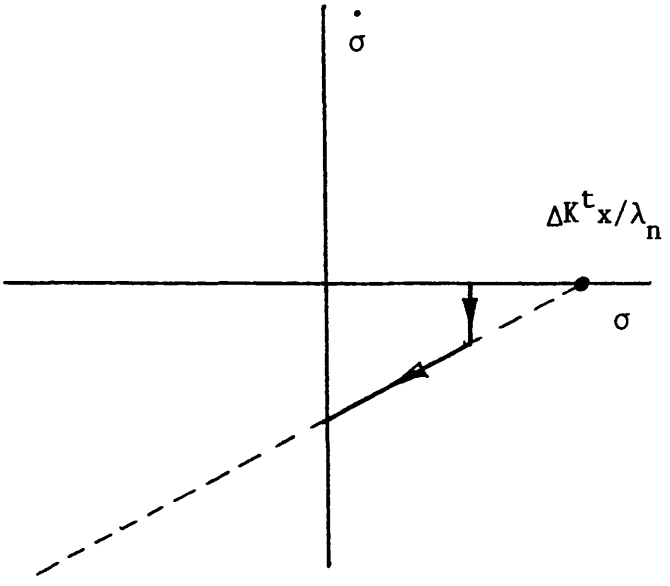
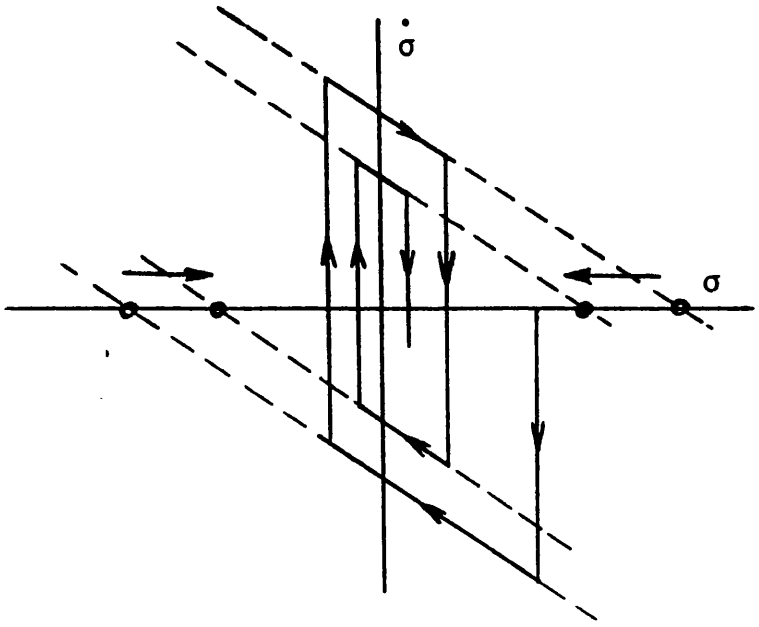


Fig. 87b As for 87a, but with λ_n unstable. ($\Delta K^t x/\lambda_n$ must be $> \sigma$).



Penetration either side of $\sigma = 0$ is very small, and depends only on the time taken to switch structures.

Fig. 88 Straight line approximation of range space trajectory for full order σ, showing convergence on origin.

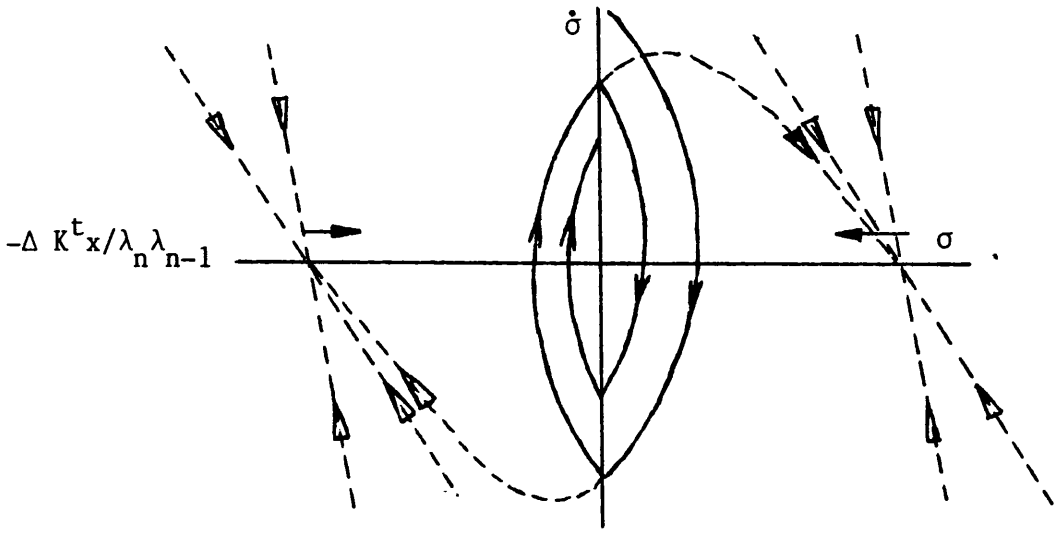


Fig. 89 Hyperbolic approximations of range space trajectories for reduced order σ with two real, stable eigenvalues, showing convergence on origin.

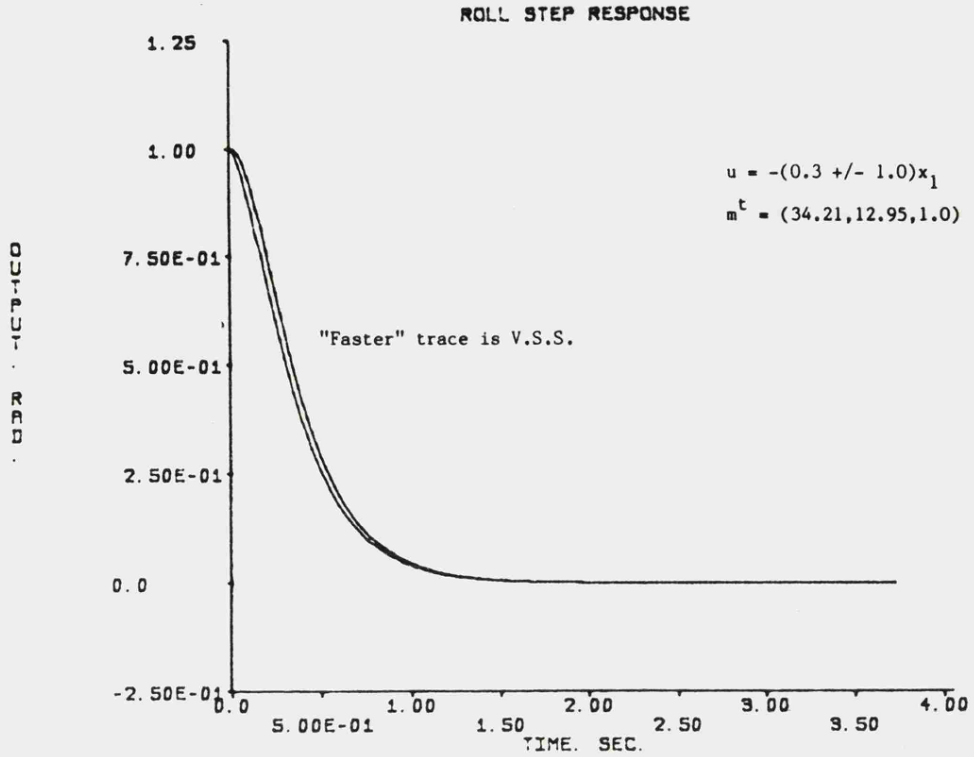


Fig. 90 Simulated step response at 40 m/s, showing quality of null space following.

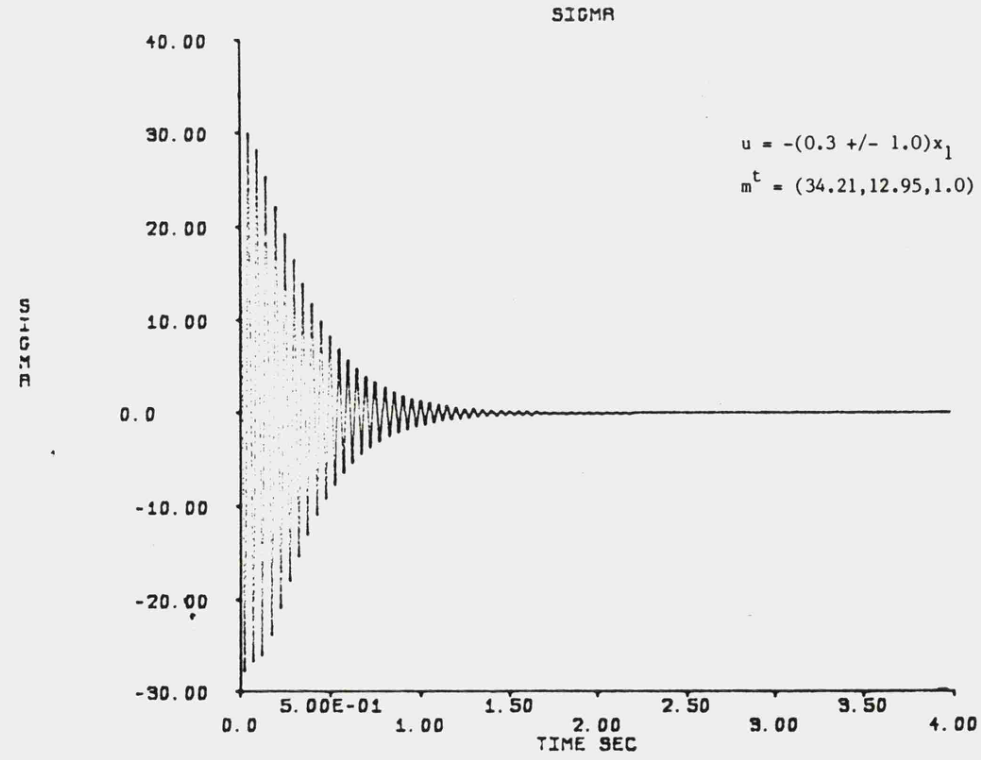


Fig. 91 Simulated full order switching function, showing effect of sampled control signal.

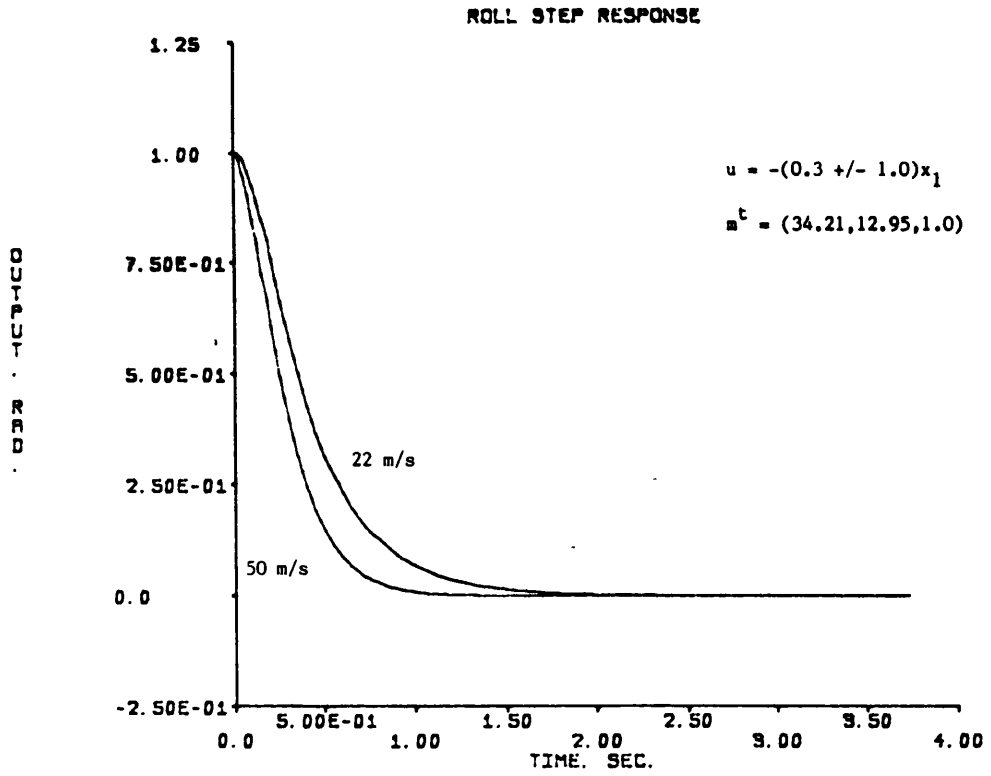


Fig. 92 Simulated step responses at 22 and 50 m/s with full order switching function ,(but not full state switching structure).

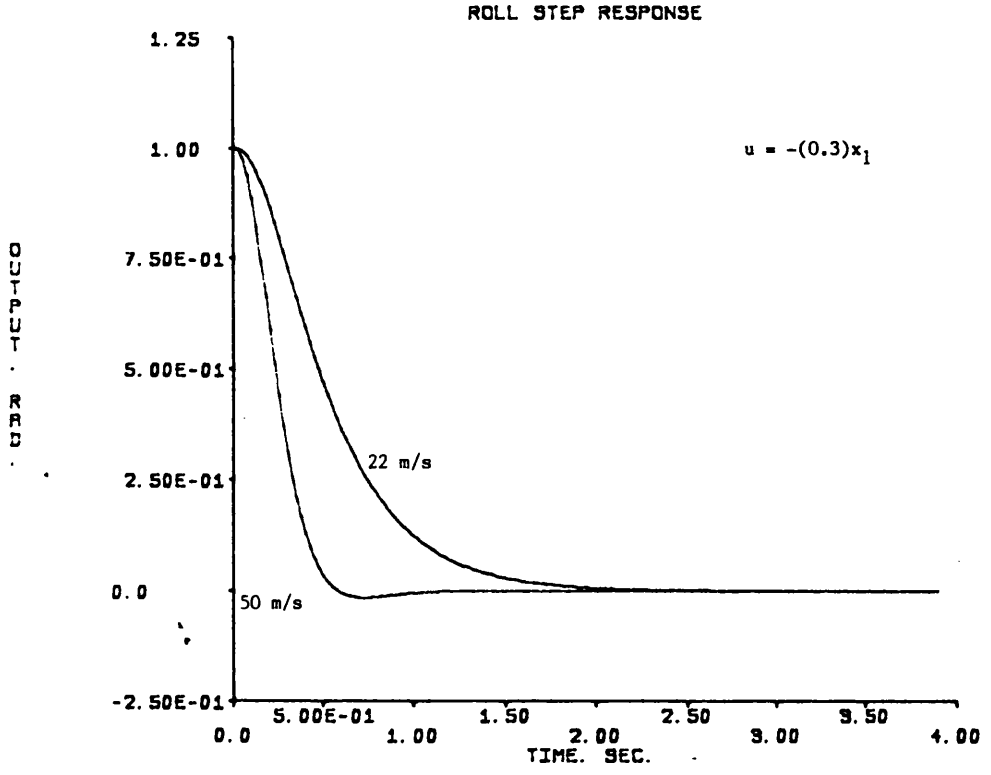
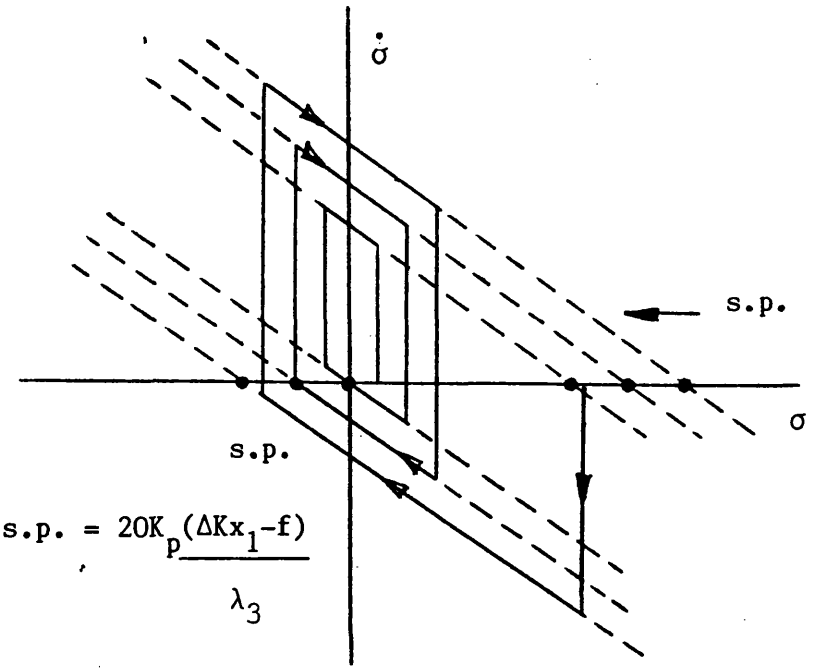


Fig. 93 Simulated step responses at 22 and 50 m/s with a fixed gain, (0.4).



Penetration either side of $\sigma = 0$ is very small, and depends only on the time it takes to switch structures.

Fig. 94 Straight line approximation of range space trajectories for full order σ, with offset stationary points.

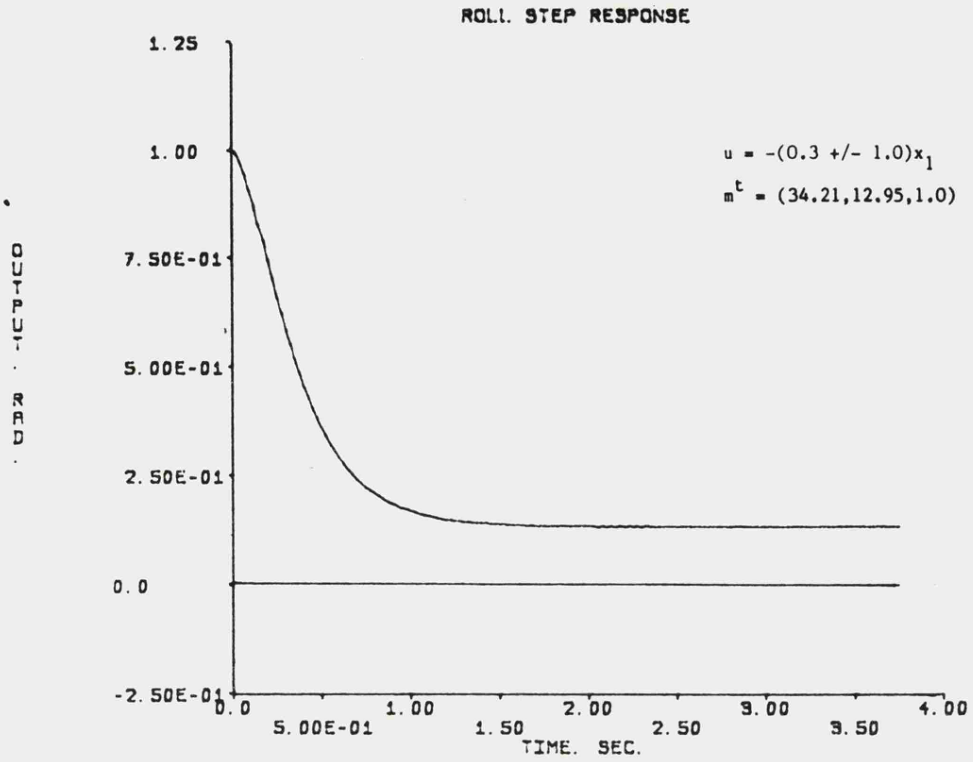


Fig. 95 Simulated step response at 40 m/s with full order switching function, and a disturbance input, (0.04 rad.).

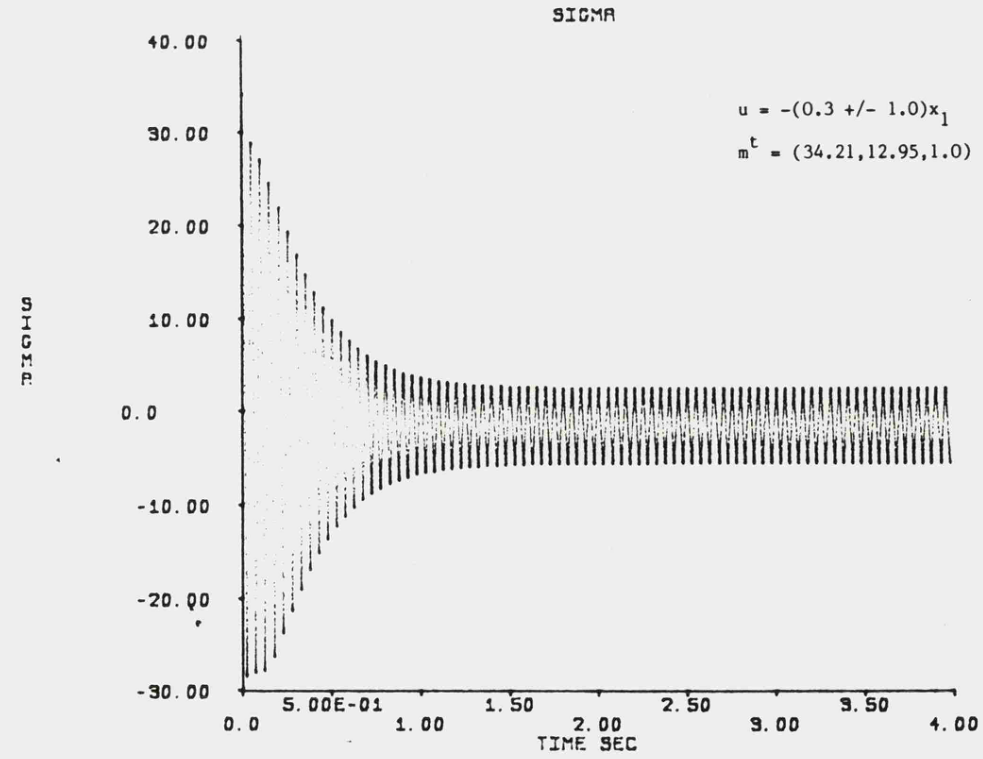


Fig. 96 Simulated full order switching function in presence (0.04rad.), showing the effect of the sampled control signal.

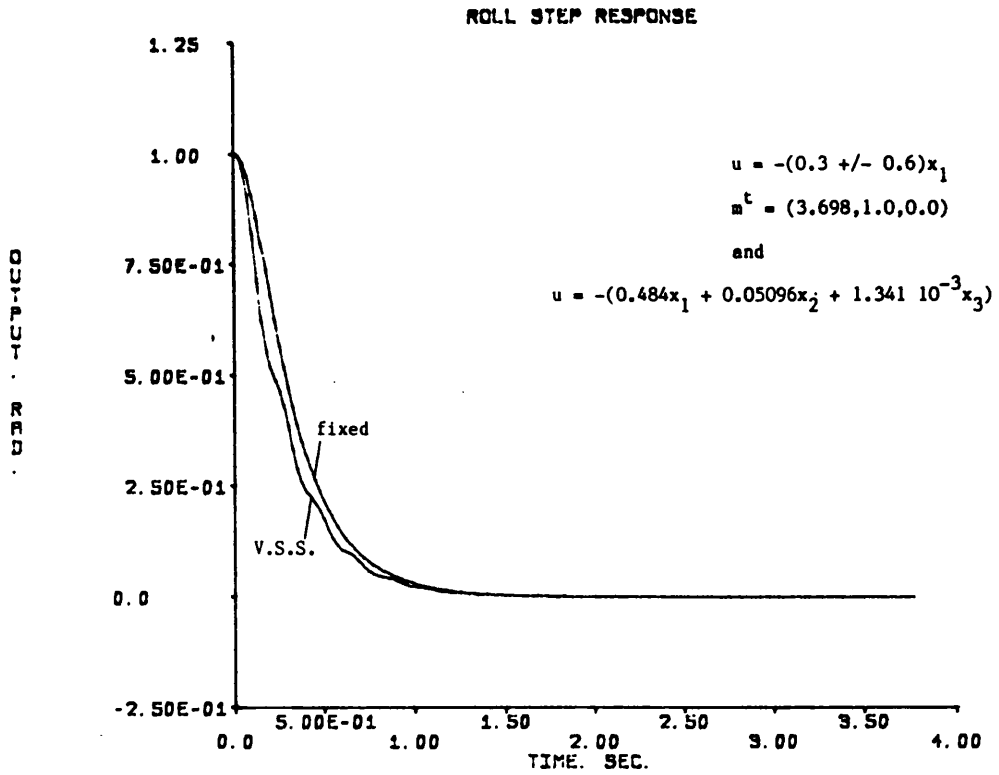


Fig. 97 Simulated step response at 40 m/s showing quality of null space following for reduced order switching function.

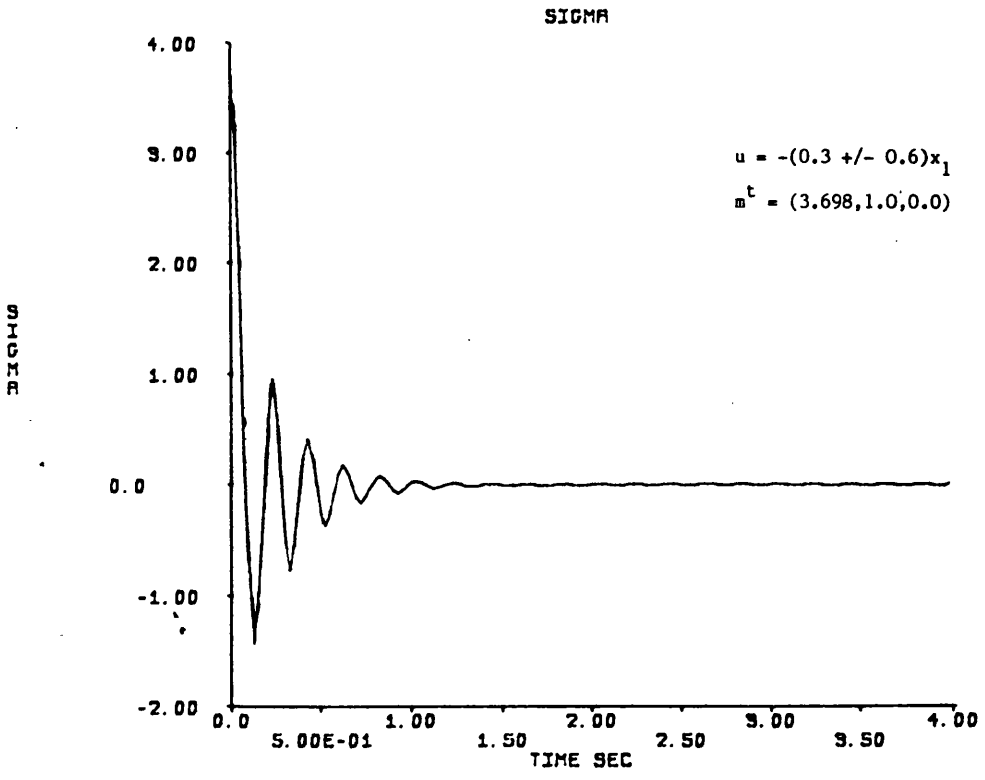


Fig. 98 Simulated reduced order switching function at 40m/s.

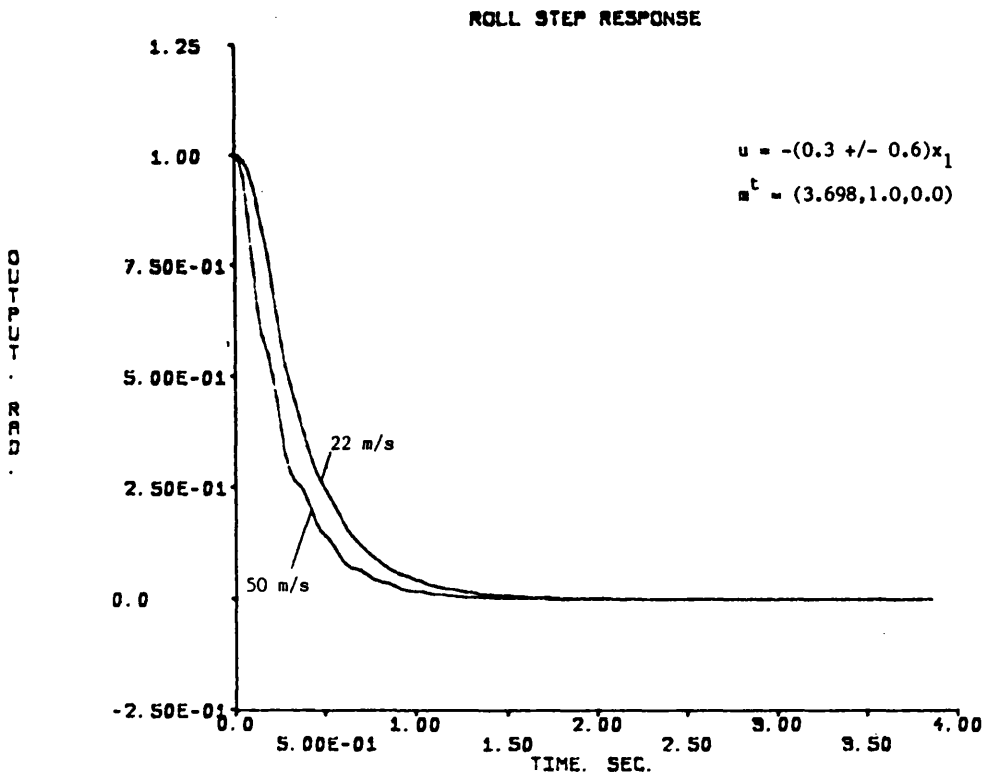


Fig. 99 Simulated step responses at 22 and 50 m/s with a reduced order switching function.

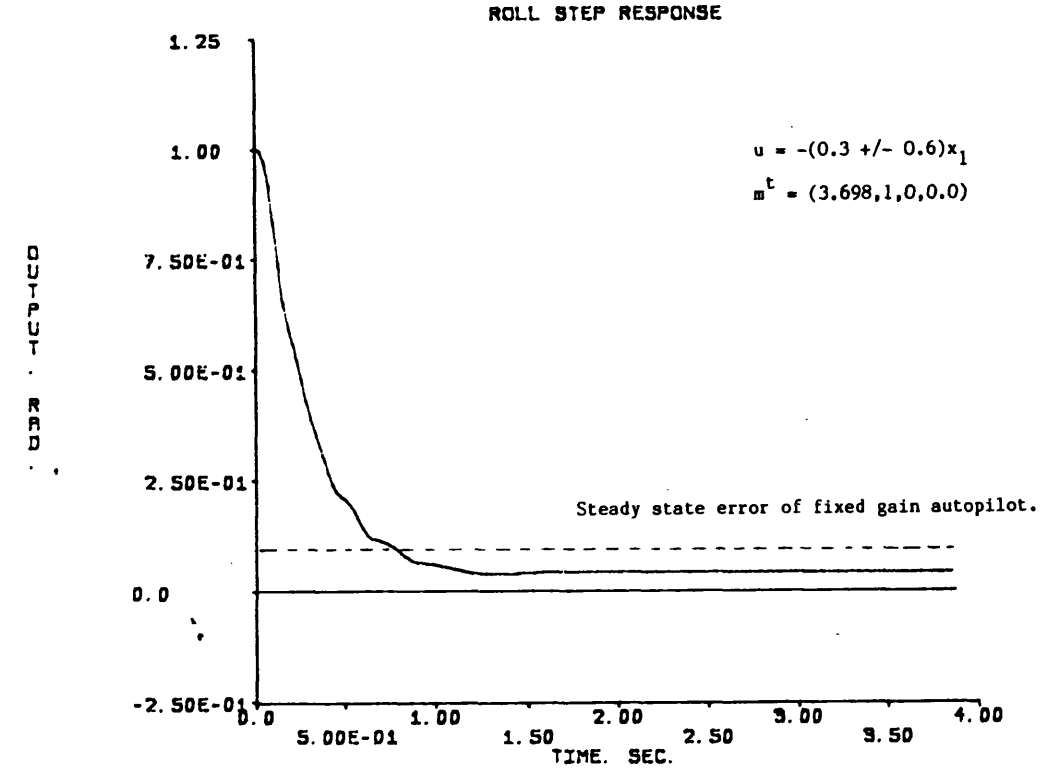


Fig. 100 Simulated step response at 40 m/s with a reduced order switching function and a disturbance input (0.04rad.).

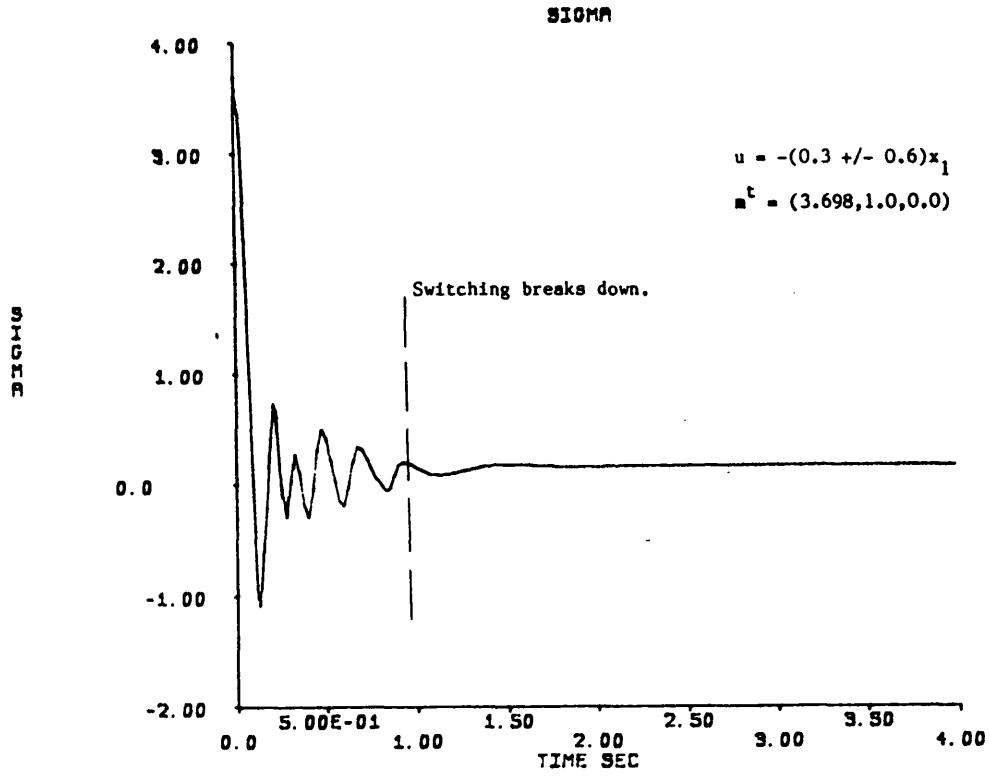


Fig. 101 Simulated reduced order switching function at 40 m/s with a disturbance input (0.04rad.).

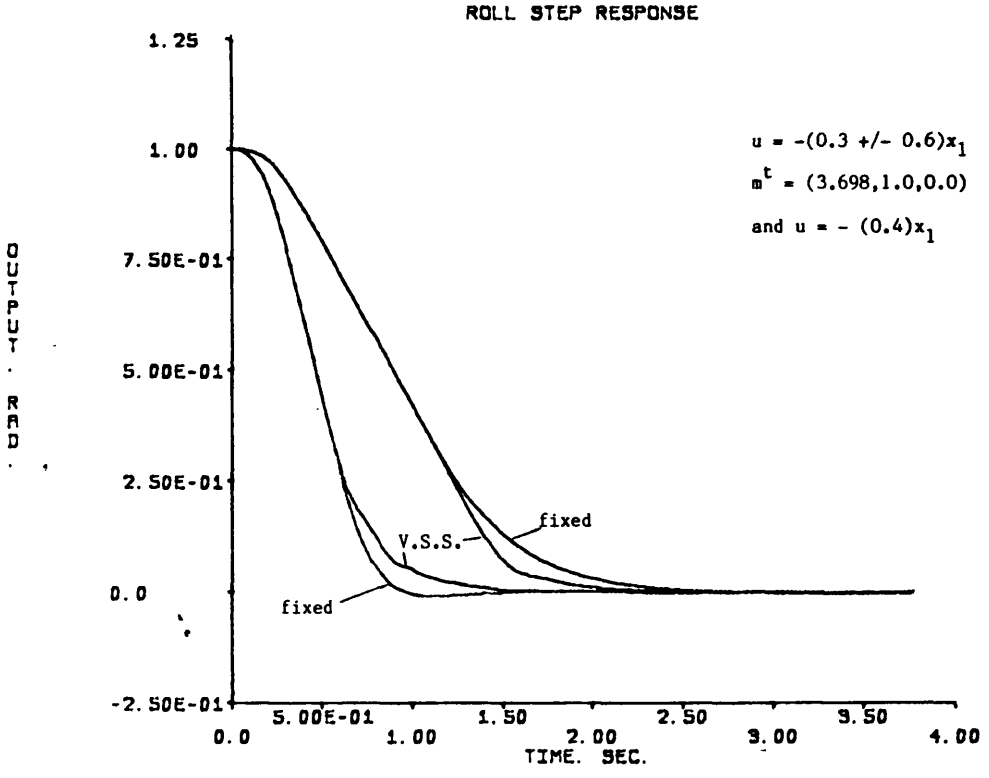


Fig. 102 Simulated step responses at 22 and 50 m/s for variable and fixed gain systems with a non-linear actuator model.

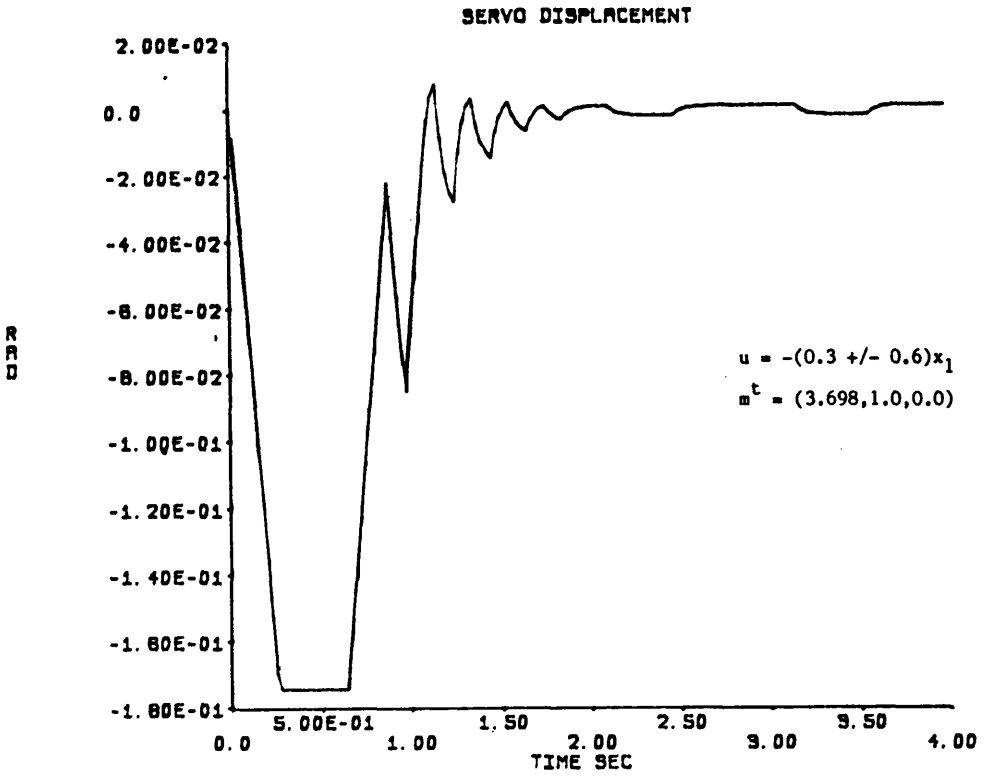


Fig. 103 Simulated servo-actuator displacement at 40 m/s for a non-linear actuator.

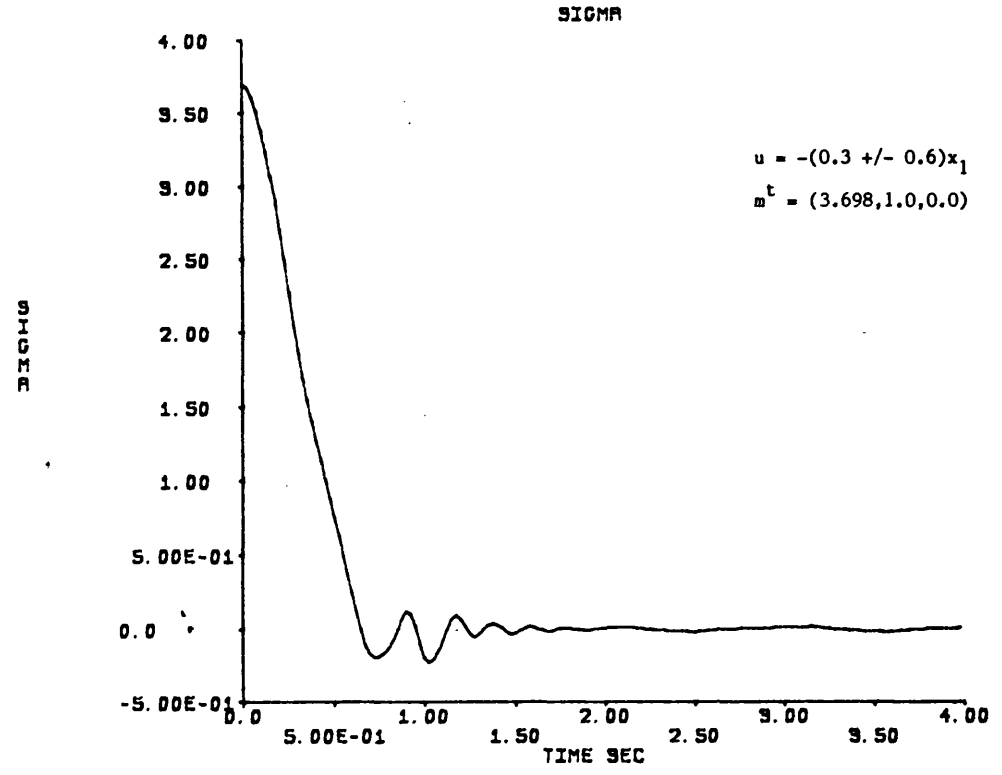


Fig. 104 Simulated reduced order switching function at 40 m/s showing effects of a non-linear actuator.

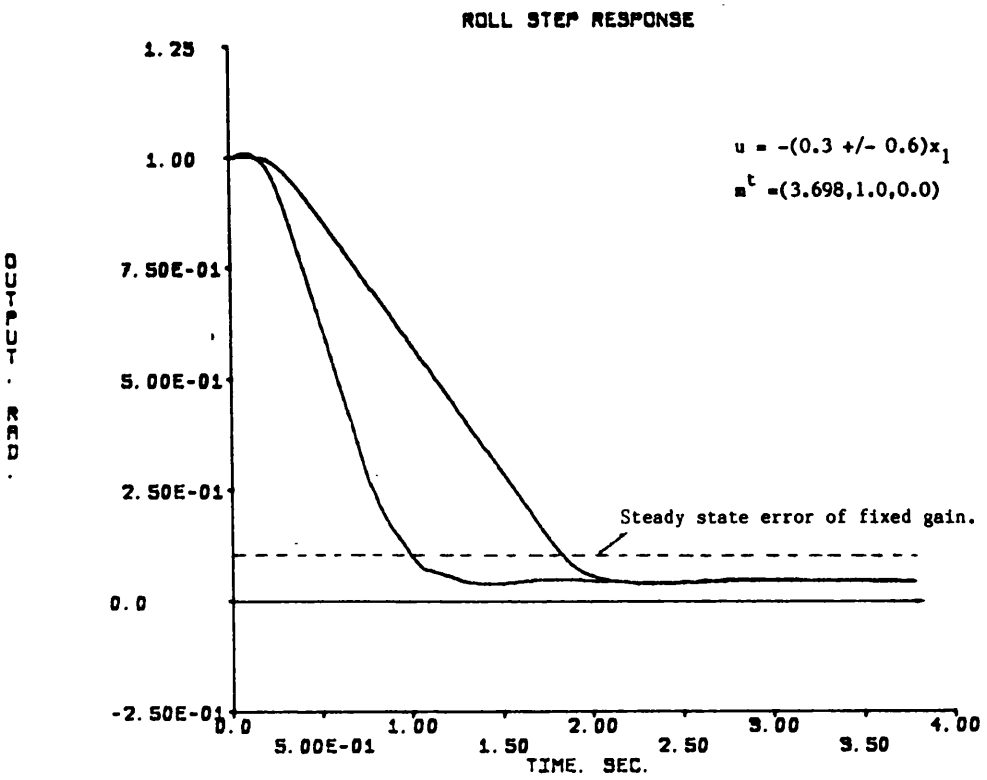


Fig. 105 Simulated step responses at 22 and 50 m/s with a reduced order switching function, non-linear servo-actuator, and a disturbance input (0.04rad.).

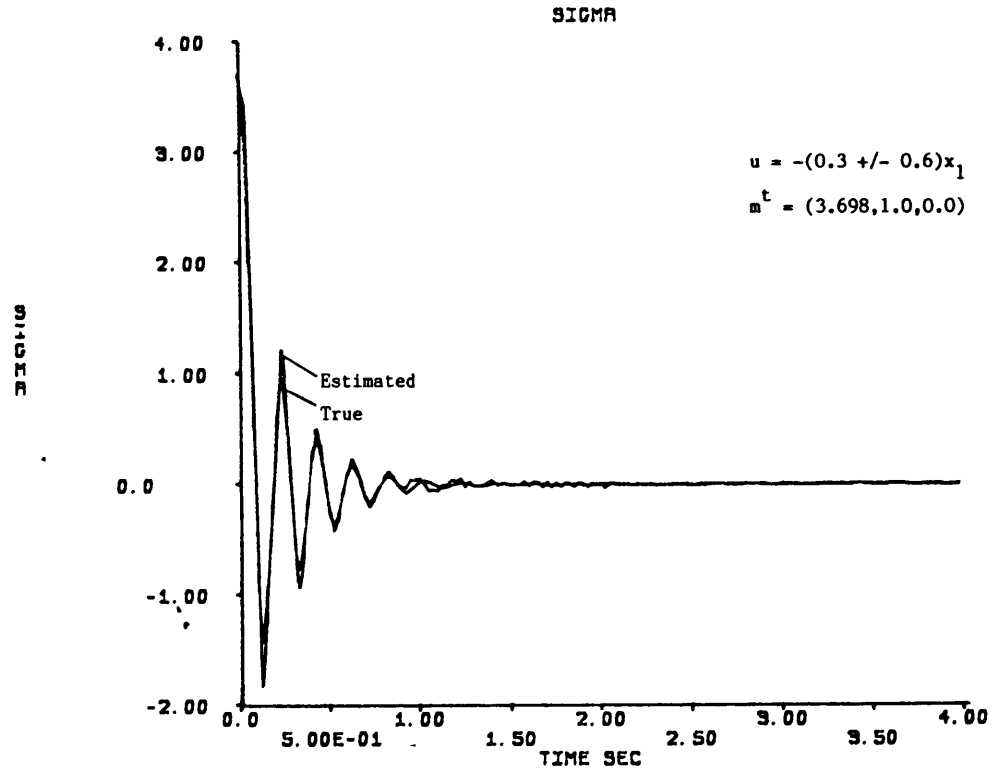


Fig. 106 Simulated reduced order switching function at 40 m/s using "true" and "estimated" rate.

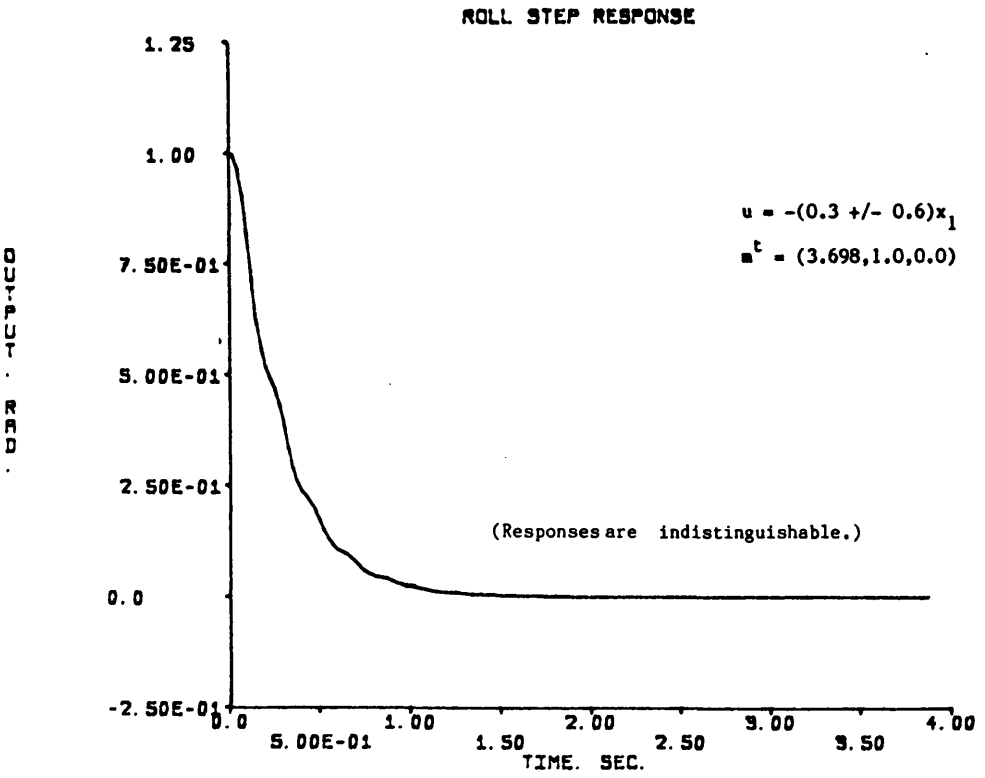


Fig. 107 Simulated step responses at 40 m/s using "true" and "estimated" rate in the switching function.

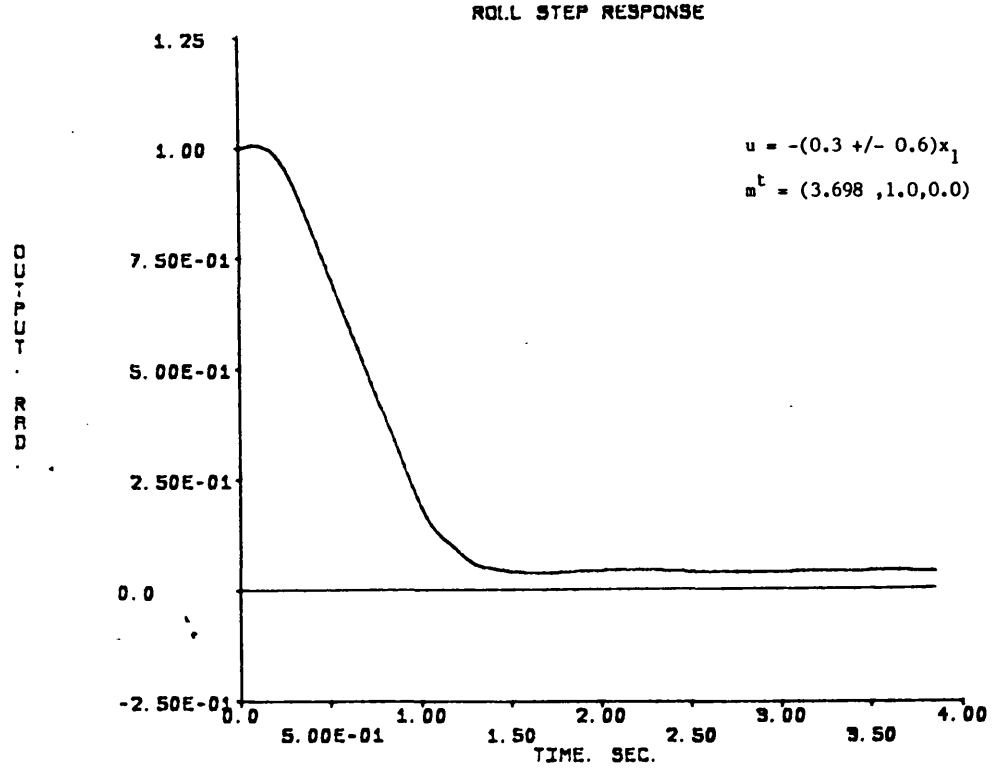


Fig. 108 Simulated step response at 40 m/s with a disturbance input and realisable controller.

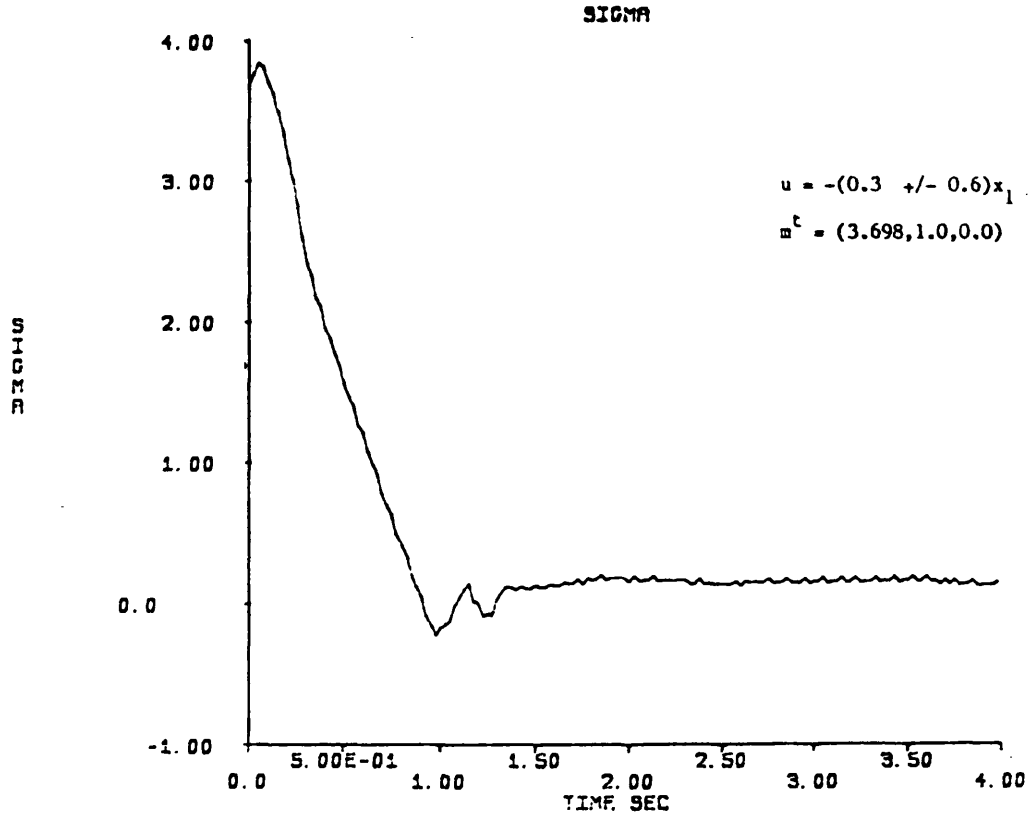


Fig. 109 Simulated reduced order switching function for step response at 40 m/s, with a realisable controller.

APPENDICES.

Appendix A.

Definition of Telemetry Channels for the DFCS.

There are 32 telemetry channels, 24 analogue, 6 digital and 2 are used for synchronisation of the receiver. Of these, approximately half are left free for payload use.

The analogue channels accept analogue input signals in the range 0.0 to 5.0V, and are specified as follows: (The upper case letters used in the function names indicate abbreviations by which the function may also be called).

Channel	Function	Status
An. 1	Metron ARMED	0V=safe, 5V=armed.
An. 3	Fuel warning	Not used.
An. 5	AIL'n DEM'd	0V=st'bd. up, 5V=st'bd. dn.
An. 7	Airspeed	0.625V=17m/s, 4.375V=53m/s.
An. 9	Yaw rate	0V=st'bd 0.8726 rad/s, 5V=pt 0.8726 rad/s
An. 11	Roll	0V=pt 1.3961 rad, 5V=st'bd 1.3961 rad.
An. 13	Pitch	0V=dn 0.6981 rad, 5V=up 0.6981 rad.
An. 15	HT ERRor	0V=7.5m low, 5V=7.5m high.
An. 17	HT. ABS'te	0V=1000.0m, 5V=0.0m.
An. 19	THR'le DEMd	0V=open, 5V=closed.
An. 21	RUD'r DEM'd	0V=st'bd yaw, 5V=pt yaw.
An. 23	EL'r DEM'd	0V=up el'tor, dn el'tor.

The digital channels are specified as follows:

DIG. 1 R.P.M.

The output from the r.p.m. counter. This is callibrated as 0= 0 r.p.m. 255= 10 000 r.p.m.

DIG. 2 TLPORT

The software generated status flags.

MSB bit 7 Heading Hold (1= engaged)
bit 6 Height Lock (1= engaged)
bit 5 Autopilot Engaged (1= engaged)
bit 4 A/D Error (1= error)
bit 3 Servo Error (1= error)
bit 2 Overflow Error (1= error)
bit 1 Above Height (1= above demanded height)
bit 0 Software Fail Flag (1= failsafe active)

DIG. 3 ACFLAGS

The hardware generated status flags.

MSB bit 7 HI
bit 6 HI
bit 5 BATOK (1= battery mains valid)
bit 4 Reset (0= reset condition)
bit 3 Squelch (1= signal present)
bit 2 Controller Select (1= digital)
bit 1 Fail Flag (0= no analogue fail)
bit 0 DIG FAIL FLAG (0= digital system error)

APPENDIX B.

EXAMPLE OF THE ASSEMBLER LISTING: THE PITCH COMPENSATOR ROUTINE.

This is an example of the output of the MICT99 assembler programme. The machine code is that which is loaded into the microprocessor's memory. The assembler code is a convenient, recognisable representation of this.

```
MACHINE CODE.       ASSEMBLER CODE.    COMMENT FIELD.
0592 0730 020C      LI    R12,SSGN     LOAD COEFF POINTER
     0732 085C'
0593 0734 020A      LI    R10,>1       SET UP SHIFT COUNT
     0736 0001
0594 0738 06A0      BL    @MULT        PERFORM STEADY STATE GAIN
     073A 04FA'
0595 073C C0C8      MOV   R8,R3        TEMP STORE
0596 073E C1C8      MOV   R8,R7        R7 = GN*FB
0597 0740 C204      MOV   R4,R8        FETCH X1
0598 0742 020A      LI    R10,>1       SET UP SHIFT COUNT
     0744 0001
0599 0746 06A0      BL    @MULT        R8 = -B1*X1  AS B1 IS NEGATIVE
     0748 04FA'
0600 074A 06A0      BL    @SATAD       R8 = GN*FB - B1*X1
     074C 054E'
0601 074E C1C8      MOV   R8,R7        TEMP STORE
0602 0750 C205      MOV   R5,R8        FETCH X2
0603 0752 020A      LI    R10,>0       SET SHIFT COUNT
     0754 0000
0604 0756 06A0      BL    @MULT        R8 = B2*X2
     0758 04FA'
0605 075A 0508      NEG   R8           R8 = -B2*X2
0606 075C 06A0      BL    @SATAD       R8 = GN*FB -B1*X1 -2*X2
     075E 054E'
0607 0760 C008      MOV   R8,R0        TEMP STORE X11
0608 0762 C1C3      MOV   R3,R7        FETCH GN*FB
0609 0764 C204      MOV   R4,R8        FETCH X1
0610 0766 020A      LI    R10,>0       SET UP SHIFT COUNT
     0768 0000
0611 076A 06A0      BL    @MULT        R8 = -(A1-B1)*X1
     076C 04FA'
0612 076E 0508      NEG   R8           R8 = (A1-B1)*X1
0613 0770 06A0      BL    @SATAD       R8 = GN*FB +(A1-B1)*X1
     0772 054E'
0614 0774 C1C8      MOV   R8,R7        TEMP STORE
0615 0776 C205      MOV   R5,R8        FETCH X2
0616 0778 020A      LI    R10,>0       SET UP SHIFT COUNT
     077A 0000
0617 077C 06A0      BL    @MULT        R8 = (A2-B2)*X2
     077E 04FA'
0618 0780 06A0      BL    @SATAD       R8 = GN*FB +(A1-B1)*X1 +(A2-B2)*X
     0782 054E'
0619 0784 C144      MOV   R4,R5        X21 BECOMES X2
0620 0786 C100      MOV   R0,R4        X11 BECOMES X1
0621 0788 0508      NEG   R8           R8 = -(MODIFIED FEEDBACK)
```

Appendix C

Sensors used in the Flight Control System.

1) Humphrey vertical axis gyroscope. VG24 04211.

A dual axis, attitude sensing gyroscope used for pitch and roll measurement, (about OX and OY).

Range:

Mechanical, +/-40 degrees (min) pitch, +/-80 degrees (min) roll.

Electrical, +/-40 (+/-2) degrees pitch, +/-80 (+/-2) degrees roll.

Electrical requirements:

+12 V, 500 mA running, (1.4 A starting current) for the motor. Output potentiometers (5k.ohm) supplied with +/-5 V for roll, and +/-12 V for pitch.

Performance:

Resolution = 0.2% of full scale.

Linearity = +/-1% of full scale.

Vertical accuracy = +/- 0.5 degrees.

Time to erect to within 1.5 degrees = 9 minutes.

2) Smiths Instruments rate gyroscope. 900 series.

Rate gyroscope used to measure rate of rotation about the aircraft OZ axis.

Range:

Mechanical, +/-50 degrees/s.

Electrical, 60, (+/-10) mV/deg/s.

Electrical requirements:

+/-6 V, 50 mA for the motor, supplies for the output signal derived from this.

Performance:

Resolution = 0.01% of full scale.

Linearity = 2% of full scale.

Time to erect = 20 seconds.

3) National Semiconductor absolute pressure sensor. LX1601A.

Provides a measure of the absolute atmospheric pressure, which is

then compared with a stored zero to indicate height.

Range:

Pressure, 10 to 20 p.s.i. (Standard atmosphere approx. 14 p.s.i.)

Electrical output, 1.0 V/p.s.i., offset at 10 p.s.i. = 2.5 V.

Electrical requirements:

+12 V, 50 mA.

Performance:

Output scaling 1.0 V/p.s.i.

Linearity = +/-0.05 p.s.i. over full range.

4) Rosemount Indicated airspeed transducer. 1221D.

Gives indication of airspeed for telemetry purposes only. Pitot static tube and ASI mounted in the starboard wing.

Range:

Airspeed, 30 to 130 knots, (15.4 to 66.9m/s).

Expected output range, 0 to +5V.

Electrical requirements:

+/-15 V, 25 mA.

Performance:

Output scaling = 50 mV/knot.

Relative accuracy, error = 0.13% of actual at 30 knots.

Appendix D.

Definition of Registers in System software.

WP0 Servo workspace.

Reg.	Label.	Function.
R0	WP0	Spare servo register.
R1	WP01	Spare servo register.
R2	WP02	Spare servo register.
R3	WP03	Aileron servo register.
R4	WP04	Rudder servo register.
R5	WP05	Elevator servo register.
R6	WP06	Throttle servo register.
R7	WP07	Check servo register.
R8	—	Servo cycle counter.
R9	—	Next servo output value.
R10	WP0A	New cycle flag.
R11	—	—
R12	—	C.R.U. base address.
R13	—	Stores WP during context switch.
R14	—	Stores ST during context switch.
R15	—	Stores PC during context switch.

WP1 Oneshot workspace.

Reg.	Label.	Function.
R0	WP0,PFLAG	Parachute request accepted.
R1	—	Timer/counter for P. Dep. routine.
R2	—	Timer/counter for Ig. cut routine.
R3	—	Timer/counter for airbag routine.
R4	—	P. Deploy procedure complete.
R5	—	Timer/counter for P. Jet. routine.
R6	—	Command word (temporary store).
R7	—	—
R8	—	—
R9	—	—

R10	—	—
R11	—	—
R12	—	C.R.U. base address.
R13	—	Stores WP during context switch.
R14	—	Stores ST during context switch.
R15	—	Stores PC during context switch.

WP2 Command workspace.

Reg.	Label.	Function.
R0	WP2	Null command.
R1	ROLDEM	Aileron demand, and calcs.
R2	PITDEM	Elevator demand.
R3	YAWDEM	Rudder demand.
R4	THRDEM	Throttle demand.
R5	—	Spare demand.
R6	—	Spare demand.
R7	—	Spare demand.
R8	—	Calcs.
R9	—	Calcs.
R10	COMMD	Command word (permanent).
R11	—	Return register.
R12	—	C.R.U. base address.
R13	—	—
R14	—	—
R15	—	—

WP3 Telemetry workspace.

Reg.	Label.	Function.
R0	WP3	Calcs for T/C routine.
R1	—	Calcs for T/C routine.
R2	TLP6 (M.s.b.)	Aileron demand for telemetry.
	TLP5 (L.s.b.)	Throttle demand for telemetry.
R3	TCKNT	Telecommand delay counter.
R4	TLP2,TLP1	TLPORT, software generated flags.
R5	FCOUNT	Timer/counter for failsafe routine.

R6	—	Servo check routine flag.
R7	TLP4 (M.s.b.)	Elevator demand for telemetry.
	TLP3 (L.s.b.)	Rudder demand for telemetry.
R8	—	Calcs.
R9	—	Calcs.
R10	—	Calcs.
R11	—	Return register.
R12	—	C.R.U. base address.
R13	—	Stores WP during context switch.
R14	—	Stores ST during context switch.
R15	—	Stores PC during context switch.

WPR Roll workspace.

Reg.	Label.	Function.
R0	WPR	Calcs.
R1	—	(K-1)th error.
R2	—	(K)th error.
R3 to R5 unused.		
R6	—	Command word (temporary store).
R7 to R10 used for calculations.		
R11	—	Return register.
R12	—	Calcs.
R13	—	Negative flag.
R14	—	—
R15	—	—

WPY Yaw workspace.

Reg.	Label.	Function.
R0	WPY	—
R1 to R5 unused.		
R6	—	Command word (temporary store).
R7 to R10 used for calculations.		
R11	—	Return register.
R12	—	Calcs.
R13	—	Negative flag.

R14	—	—
R15	—	—

WPP Pitch workspace.

Reg.	Label.	Function.
R0	WPP	Digital filter state(X11).
R1	—	(K-1)th error.
R2	—	(K-2)th error.
R3	—	(K)th error.
R4	—	Digital filter state (X1).
R5	—	Digital filter state (X2).
R6	—	Temporary Command register.
R7 to R10 used for		calculations.
R11	—	Return register.
R12	—	Calcs.
R13	—	Negative flag.
R14	—	—
R15	—	—

WPH Height workspace.

Reg.	Label.	Function.
R0	WPH,HTPIT	Autopilot derived height demand.
R1	—	Height zero store.
R2	—	Shuffled complement of above.
R3	— (M.s.b.)	Height error for telemetry.
	TLP7 (M.s.b.)	Height absolute for telemetry.
R4	—	—
R5	HTFB	Height feedback.
R6	—	Command word (temporary store).
R7 to R10 used for		calculations.
R11	—	Return register.
R12	—	Calcs.
R13	—	Negative flag.
R14	—	—
R15	—	—

APPENDIX E

PROGRAMMES USED AS ARITHMETIC AIDS IN THE Z-TRANSFORMATION OF THE AIRCRAFT TRANSFER FUNCTIONS.

E.1 SUMMARY.

Both programmes run on the mainframe computing facility at Bath University. This is so that they can make use of a suite of numerical algorithms, designed to be called as subroutines from the user's programme. ZTRAN.FORTRAN calls F04ATF, which solves a set of simultaneous equations presented in matrix form, and C02AEF which finds the roots of a real polynomial equation. GENLOC.FORTRAN only calls C02AEF. Full details of these are given in reference (E1). Listings of ZTRAN.FORTRAN and GENLOC.FORTRAN are given in sections E.4 and E.5.

E.2 ZTRAN.FORTRAN, the z-transformation routine.

It must be noted that this is only intended as an aid to calculation. In order to obtain the correct result, the input must be entered in a given manner, which dictates the nature of the output. The presence of a zero-order-hold is automatically assumed, so the input takes the form:

$$\frac{(z-1)}{z} \mathcal{Z}\left(\frac{N(s)}{sD(s)}\right) \qquad \text{E1}$$

N(s) is a polynomial in s, of order < 6, and sD(s) is one of three different types of function, as follows:

TYPE 1 $sD(s) = s^2((s+b)^2 + c^2)((s+d)^2 + e^2)$

TYPE 2 $sD(s) = s^2(s+b)(s+c)((s+d)^2 + e^2)$

TYPE 3 $sD(s) = s^2(s+b)(s+c)(s+d)(s+e)$

These are the only functions that will be accepted, so the input must be arranged to comply. If the order of the denominator is less than 6, the function must be expanded by introducing extra factors to the numerator and denominator alike. An example of

this method will be given later.

The treatment of the input types differs in detail, but the general process is the same, so consider a type 1 input. The function to be transformed is:

$$F(s) = \frac{N(s)}{sD(s)}, \text{ so } F(s) =$$

$$\frac{N(0) + N(1)s + N(2)s^2 + N(3)s^3 + N(4)s^4 + N(5)s^5}{s^2((s+b)^2 + c^2)((s+d)^2 + e^2)} \qquad E2$$

If the function has non-unity forward gain K, this is introduced as a common factor in N(0),N(1)...N(5). Furthermore, if the order of the numerator is less than 5, N(5),N(4) etc. must be entered as 0.0.

The first stage is to separate this into partial fractions, in order that the z-transformation can be carried out:

$$\text{So, } F(s) = \frac{L}{s} + \frac{M}{s^2} + \frac{Ps+Q}{((s+b)^2 + c^2)} + \frac{Rs+S}{((s+d)^2 + e^2)} \qquad E3$$

Equating the expressions in E2 and E3, and multiplying throughout by D(s) gives:

$$Ls((s+b)^2 + c^2)((s+d)^2 + e^2) + M((s+b)^2 + c^2)((s+d) + e^2) + (Ps+Q)s^2((s+d)^2 + e^2) + (Rs+S)s^2((s+b)^2 + c^2) = N(0) + N(1)s + N(2)s^2 + N(3)s^3 + N(4)s^4 + N(5)s^5 \qquad E4$$

By gathering the terms in s^0, s^1, s^2, s^3, s^4 and s^5, a set of simultaneous equations can be formed, which can be written in matrix form, to give:

$$\mathbf{A}\,\underline{x} = \underline{b} \qquad E5$$

where $\underline{x}^t = (L, M, P, Q, R, S)$

and $\underline{b}^t = (N(0), N(1), N(2), N(3), N(4), N(5))$

and $\mathbf{A}$ is a sixth order matrix, whose elements are functions of b,c,d and e.

The first activity of the programme is to establish the **A** matrix for the particular input function. There are three configurations of **A** depending on the type of denominator. The subroutine F04ATF is then called to solve equation E5, and evaluate L,M,P,Q,R and S. Knowing these, the individual transformations of the functions in E3, can be carried out, to give:

$$\frac{(z-1)}{z}\mathcal{Z}(F(s)) = \frac{z-1}{z}\frac{Lz}{(z-1)} + \frac{MTz}{(z-1)^2} + \frac{Pz(z+f)}{z^2+gz+h} + \frac{Rz(z+j)}{z^2+kz+m} \qquad E6$$

where T is the sampling period,
and f,g,h and j,k,m are obtainable from tables.

As explained in the main text (section 5.4.2.1), this result is more useful when expressed as a product of factors, rather than a sum of parts. This requires that the function in E6 be re-arranged to give:

$$\frac{(z-1)}{z}\mathcal{Z}(F(s)) = \frac{H(z)}{(z-1)(z^2+gz+h)(z^2+kz+m)} \qquad E7$$

By the reverse process of that described earlier, H(z) is derived as:

$$H(z) = P(0) + P(1)z + P(2)z^2 + P(3)z^3 + P(4)z^4 + P(5)z^5 \qquad E8$$

where P(0), P(1)....P(5) are functions of the constants L,M,P Q,R,S and f,g,h,j,k,m.

The programme evaluates P(0)...P(5) and then calls the subroutine C02AEF to find the roots of the polynomial in E8. The results of the transformation process are presented in two parts; the numerator is given as a gain, K, in conjunction with the roots of H(z), and the denominator can have one of three forms, depending on the type of input. These are:

TYPE 1 $(z-1)(z^2+gz+h)(z^2+kz+m)$

TYPE 2 $(z-1)(z+g)(z+h)(z^2+kz+m)$

TYPE 3 $(z-1)(z+g)(z+h)(z+k)(z+m)$

An example will be given to illustrate the use of ZTRAN.FORTRAN Consider the z-transformation of the 40m/s roll transfer function with a zero-order-hold:

$$G_R(z) = \frac{(z-1)}{z} \mathcal{Z}\left(\frac{(-152.8)}{s^2(s+19.61)}\right) \qquad \text{E9}$$

This does not have one of the necessary denominator types, so the numerator and denominator are given extra, identical factors to suit. These cancel each other out in the calculations and do not effect the result. To facilitate this, the programme checks the values returned by subroutine F04ATF, and if they are less than $1x10^{8}$, they are set to zero. This takes care of in-exact cancellation caused by lack of precision in the algorithm. So the equation in E9 is re-arranged to give a type 2 input, as follows:

$$G_R(z) = \frac{(z-1)}{z} \mathcal{Z}\left(\frac{(-152.8(s+1)((s+2)^2+3^2))}{s^2(s+19.61)(s+1)((s+2)^2+3^2)}\right) . \qquad \text{E10}$$

then,

$$G_R(z) = \frac{(z-1)}{z} \mathcal{Z}\ (-)\left(\frac{1986.4+2597.6s+764.0s^2+152.8s^3+0.0s^4+0.0s^5}{s^2(s+19.61)(s+1)((s+2)^2+3^2)}\right) \qquad \text{E11}$$

This is entered into the programme in the following manner:

```
PROMPT  ENTER DENOMINATOR TYPE 1,2 OR 3
INPUT   2
PROMPT  ENTER b,c,d,e
INPUT   19.61,1.0,2.0,3.0
PROMPT  ENTER NUMERATOR N(0),N(1),N(2),N(3),N(4),N(5)
INPUT   1986.4,2597.6,764.0,152.8,0.0,0.0
PROMPT  ENTER SAMPLING INTERVAL
INPUT   0.025
```

The result of the transformation is given as follows:

```
Z TRANSFORM IS K*N(Z)/D(Z), WHERE K = 0.408166 D-01
ROOTS OF NUMERATOR ARE:
REAL PART = -0.8495 D00        IMAG PART = 0.0 D00
REAL PART = 0.0 D00            IMAG PART = 0.0 D00
THE DENOMINATOR IS =
(Z-1)(Z-0.6125 D00)(Z-0.0 D00)(Z**2 + 0.0 D00Z + 0.0 D00)
```

E.3 GENLOC.FORTRAN, the root locus routine.

This programme is used to solve:

$$1.0 + KK' \frac{(a + bz + cz^2 + dz^3 + ez^4)}{(r + sz + tz^2 + uz^3 + vz^4 + wz^5)} = 0.0 \qquad \text{E12}$$

as K varies, making use of the subroutine C02AEF, as in ZTRAN.FORTRAN. An example will be given to illustrate the use of GENLOC.FORTRAN. Consider the 40m/s roll aircraft transfer function, with the small deviation linear servo model. This has been transformed using ZTRAN.FORTRAN to give:

$$G_{RSV}(z) = \frac{0.625\ 10^{-2}(z^2 + 3.138z + 0.60944)}{z^3 + (-2.219)z^2 + 1.5905z + (-0.3715)} \qquad \text{E13}$$

It is necessary that the numerator and denominator be multiplied by z^2 to avoid division by zero in the calculations. The data is entered in the order:

K',a,b,c,d,e,r,s,t,u,v,w

Therefore the input for this example will be:

```
0.625 10^-2,0.0,0.0,0.60944,3.138,1.0,0.0,0.0,0.0,-0.3715,
1.5905,-2.219,1.0
```

The operator then types the desired value of K, whereupon the routine evaluates and displays the roots. The next operator input

is an integer switch, which is interpreted by the programme as:

0, stop

1, read new value of K and repeat.

E.4 Listing of ZTRAN.FORTRAN.

```
      real*8 a(10,10),b(10),c(10),am(12,12),w(18),x(18),as,bs,cs
      real*8 ds,es,t,zf,zg,zh,phi,zj,zk,zm,theta,p(6),v(5),pi,toI
      real*8 x02aaf,re(5),im(5),d1,d2,d3,d4,lim,bz,cz,dz,ez
      integer ia,n,iaa,ifail,ifal,nn,itype
      write (6,600)
600   format (1h ,'system is n(s)/d(s), order of n(s)<6,order d(s)=6')
      write (6,601)
601   format (1h ,'type 1 d(s)=s**2((s+b)**2+c**2)((s+d)**2+e**2)')
      write (6,602)
602   format (1h ,'type 2 d(s)=s**2(s+b)(s+c)((s+d)**2+e**2)')
      write (6,603)
603   format (1h ,'type 3 d(s)=s**2(s+b)(s+c)(s+d)(s+c)')
      write (6,604)
604   format (1h ,'enter denominator type 1,2 or 3')
      read (5, ) itype
      write (6,605)
605   format (1h ,'enter b,c,d,e')
      read (5, ) bs,cs,ds,es
      do 100 j=1,6
      b(j)=0.0d0
      do 110 k=1,6
      a(j,k)=0.0d0
110   continue
100   continue
      write (6,606)
606   format (1h ,'enter numerator n(0) n(1) n(2) n(3) n(4) n(5)')
      read (5, ) b(1),b(2),b(3),b(4),b(5),b(6)
      if(itype.ne.1) go to 120
      d2=(bs**2+cs**2)
      d1=2.0d0*bs
      d4=(ds**2+es**2)
      d3=2.0d0*ds
      a(6,5)=1.0d0
      a(5,6)=1.0d0
      a(5,5)=d1
      a(5,4)=1.0d0
      a(5,3)=d3
      a(4,6)=d1
      a(4,5)=d2
      a(4,4)=d3
      a(4,3)=d4
      a(3,6)=d2
      a(3,4)=d4
      go to 140
120   if(itype.ne.2) go to 130
      d2=bs*cs
```

ZTRAN.FORTRAN Listing cont.

```
      d1=bs+cs
      d4=(ds**2+es**2)
      d3=2.0d0*ds
      a(6,5)=1.0d0
      a(6,4)=1.0d0
      a(5,6)=1.0d0
      a(5,5)=d1
      a(5,4)=bs+d3
      a(5,3)=cs+d3
      a(4,6)=d1
      a(4,5)=d2
      a(4,4)=d4+bs*d3
      a(4,3)=d4+cs*d3
      a(3,6)=d2
      a(3,4)=bs*d4
      a(3,3)=cs*d4
      go to 140
130   if(itype.ne.3) go to 150
      d2=bs*cs
      d1=bs+cs
      d3=ds+es
      d4=ds*es
      a(6,6)=1.0d0
      a(6,5)=1.0d0
      a(6,4)=1.0d0
      a(5,6)=d1+ds
      a(5,5)=d1+es
      a(5,4)=bs+d3
      a(5,3)=cs+d3
      a(4,6)=d1*ds+d2
      a(4,5)=d1*es+d2
      a(4,4)=bs*d3+d4
      a(4,3)=cs*d3+d4
      a(3,6)=ds*d2
      a(3,5)=es*d2
      a(3,4)=bs*d4
      a(3,3)=cs*d4
140   a(6,3)=1.0d0
      a(6,1)=1.0d0
      a(5,2)=1.0d0
      a(5,1)=d1+d3
      a(4,2)=d1+d3
      a(4,1)=d2+d4+d1*d3
      a(3,2)=d2+d4+d1*d3
      a(3,1)=d1*d4+d3*d2
      a(2,2)=d1*d4+d3*d2
      a(2,1)=d2*d4
      a(1,2)=d2*d4
      n=6
      ia=10
      iaa=12
      ifail=1
      call f04atf (a,ia,b,n,c,aa,iaa,w,x,ifail)
      if(ifail.eq.0) go to 160
      write (6,607) ifail
```

```
607       format (1h ,"ifail = ",i1)
          stop
160       write (6,608) c(1),c(2)
608       format (1h ,'l=',d13.6,'m=',d13.6)
          write (6,609) c(3),c(4),c(5),c(6)
609       format (1h ,'p=',d13.6,'q=',d13.6,'r=',d13.6,'s=',d13.6)
          lim=0.1d-07
          write (6,610)
610       format (1h ,'enter sampling interval T')
          read (5, ) t
          if(dabs(c(2)).le.lim) c(2)=0.0d0
          c(2)=t*c(2)
          if(itype.ne.1) go to 170
          if(dabs(c(3)).le.lim) go to 180
          phi=datan((bs-c(4)/c(3))/cs)
          zf=-1.0d0*dexp(-1.0d0*bs*t)*(dcos(cs*t-phi))/dcos(phi)
          zg=-2.0d0*(dexp(-1.0d0*bs*t))*dcos(cs*t)
          zh=dexp(-2.0d0*bs*t)
          go to 190
180       c(3)=0.0d0
          zf=0.0d0
          zg=0.0d0
          zh=0.0d0
190       if(dabs(c(5)).le.lim) goto 200
          theta=datan((ds-c(6)/c(5))/es)
          zj=-1.0d0*dexp(-1.0d0*ds*t)*(dcos(es*t-theta))/dcos(theta)
          zk=-2.0d0*(dexp(-1.0d0*ds*t))*dcos(es*t)
          zm=dexp(-2.0d0*ds*t)
          go to 210
200       c(5)=0.0d0
          zj=0.0d0
          zk=0.0d0
          zm=0.0d0
          go to 210
170       if(dabs(c(3)).le.lim)go to 220
          bz=dexp(-1.0d0*bs*t)
          go to 230
220       c(3)=0.0d0
          bz=0.0d0
230       if(dabs(c(4)).le.lim)go to 240
          cz=dexp(-1.0d0*cs*t)
          go to 250
240       c(4)=0.0d0
          cz=0.0d0
250       zf=-1.0d0*(c(3)*cz+c(4)*bz)/(c(3)+c(4))
          c(3)=c(3)+c(4)
          zg=-1.0d0*(bz+cz)
          zh=bz*cz
          if(itype.eq.2) go to 190
          if(dabs(c(6)).le.lim) go to 260
          dz=dexp(-1.0d0*ds*t)
          go to 270
260       c(5)=0.0d0
          dz=0.0d0
270       if(dabs(c(6)).le.lim) go to 280
          ez=dexp(-1.0d0*es*t)
```

ZTRAN.FORTRAN Listing cont.

```
        go to 290
290     c(6)=0.0d0
        ez=0.0d0
290     zj=-1.0d0*(c(5)*ez+c(6)*dz)/(c(5)+c(6))
        c(5)=c(5)+c(6)
        zk=-1.0d0*(dz+ez)
        zm=dz*ez
210     p(1)=c(1)+c(3)+c(5)
        p(2)=c(1)*(zg+zk-1.0d0)+c(2)+c(3)*(zk-2.0d0+zf)+c(5)*(zg-2.0d0+zj)
        p(3)=c(1)*(zh+zm+zg*zk-zg-zk)+c(2)*(zg+zk)
        p(3)=p(3)+(c(3)*(zm+1.0d0-2.0d0*zk+zf*zk-2.0d0*zf))
        p(3)=p(3)+(c(5)*(zh+1.0d0-2.0d0*zg+zj*zg-2.0d0*zj))
        p(4)=c(1)*(zm*zg+zk*zh-zh-zm-zg*zk)+c(2)*(zh+zm+zg*zk)
        p(4)=p(4)+(c(3)*(zk-2.0d0*zm+zf*zm+zf-2.0d0*zk*zf))
        p(4)=p(4)+(c(5)*(zg-2.0d0*zh+zj*zh+zj-2.0d0*zg*zj))
        p(5)=c(1)*(zh*zm-zm*zg-zk*zh)+c(2)*(zm*zg+zk*zh)
        p(5)=p(5)+c(3)*(zm+zf*zk-2.0d0*zf*zm)+c(5)*(zh+zj*zg-2.0d0*zj*zh)
        p(6)=c(1)*(-1.0d0*zh*zm)+c(2)*zh*zm+c(3)*zf*zm+c(5)*zj*zh
        v(1)=1.0d0
        v(2)=p(3)/p(2)
        v(3)=p(4)/p(2)
        v(4)=p(5)/p(2)
        v(5)=p(6)/p(2)
        p1=0.1d0
        tol=x02aaf(P1)
        ifal=0
        nn=5
        call c02aef (v,nn,re,im,tol,ifal)
        if (ifal.eq.0) go to 300
        write (6,620) ifal
620     format (1h ,"ifal =",i1)
        stop
300     write (6,612) p(2)
612     format (1h ,'z transform is K*n(z)/d(z), where K=',d13.6)
        write (6,613)
613     format (1h ,' Roots of the numerator are=')
        do 310 i=1,4
        write (6,614) re(i),im(i)
614     format (1h ,' Real part=',d13.6,'  Imag part =',d13.6)
310     continue
        write (6,615)
615     format (1h ,' The denominator is =')
        if(itype.ne.1) go to 320
        write (6,616) zg,zh,zk,zm
616     format (1h ,'(z-1)(z**2+',d11.4,'z+',d11.4,')(z**2+',d11.4,'z+',d11.4,')')
        go to 150
320     if(itype.ne.2) go to 330
        write (6,617) hz,dz,zk,zm
617     format (1h ,'(z-1)(z-',d11.4,')(z-',d11.4,'(z**2+',d11.4,'z+',d11.4,')')
        go to 150
330     write (6,618) hz,dz,dz,ez
618     format (1h ,'(z-1)(z-',d11.4,')(z-',d11.4,')(z-',d11.4,')(z-',d11.4,')')
150     stop
        end
```

E.5 Listing of GENLOC.FORTRAN.

```
      real*8 a(6),re(6),im(6),alpha,beta,gam,del,ep,v,w,r,s,t,u,k,kdash,p1,x02aaf,tol
      integer i,ifail,ians,n
      read (5, )kdash,alpha,beta,gam,del,ep,r,s,t,u,v,w
  120 read (5, )k
      a(1)=1.0d0
      a(2)=(v+k*kdash*ep)/w
      a(3)=(u+k*kdash*del)/w
      a(4)=(t+k*kdash*gam)/w
      a(5)=(s+k*kdash*beta)/w
      a(6)=(r+k*kdash*alpha)/w
      p1=0.1d0
      tol=x02aaf(p1)
      do 160 j=1,6
      re(j)=0.0d0
      im(j)=0.0d0
  160 continue
      ifail=0
      n=6
      call c02aef(a,n,re,im,tol,ifail)
      if(ifail.ne.0) go to 100
      do 110 j=1,6
      write (6,600)re(j),im(j)
  600 format (1h ,'real part=',d13.4,'imag part=',d13.4)
  110 continue
  130 read (5, )ians
      if (ians.eq.0) go to 140
      if (ians.eq.1) go to 120
      go to 130
  100 write (6,601)ifail
  601 format ('ifail=',i1)
  140 stop
      end
```

Reference .

(E1) Nag fortran library manual: F04ATF Nagflib 1200/646 mk10 2nd Nov 82

C02AEF " 1271/728 mk10 2nd Feb 83

Appendix F.

The digital simulation routines.

F.1 Summary

These are interactive simulations run on a PDP 11 minicomputer. They are based on z-transforms of the roll or pitch attitude loops, and use a series of difference equations to implement the z-transfer functions. The roll simulation will be described, and a listing given. The pitch simulation differs in detail, but follows the same principles.

F.2 Structure of ZROLL.FTN

There are three stages, being definition, simulation and output , perfomed one after the other. The former allows the operator to specify the switching function, the servomotor parameters, the rate type (i.e. derived from a 320.0Hz z-transfer function and considered to be "true", or from a 40.0Hz, 3 point FIR algorithm with 12 bit quantisation, and considered to be "estimated"), and the gains. A 40m/s A.T.F. is assumed for the first run, but this can be changed to 22 or 50m/s from the output menu, and then re-run.

The simulation stage simulates a 4 second period, with initial steady state conditions of error = 1.0rad. The demanded attitude is 0.0rad. The routine evaluates the switching function, and then determines the gains to be used in order to generate the servo demand. These activities are performed once every 8 iterations, simulating the 40.0Hz sampling rate of the DFCS. The rest of the simulation is performed every iteration. This includes determining the servo-actuator position

and rate limits, and then evaluating the state and output difference equations.

The output stage presents the operator with an option menu which includes changing the airspeed, the gains or the switching function, and re-running the simulation, or displaying the attitude, the servo-actuator displacement or the switching function. Hard copies can be produced.

F.3 Listing of ZROLL.FORTRAN.

```
      REAL*4 TIME(160),ROLL(160),SIGMA(160),GAINS(160)
      REAL*4 SERVO(160)
      LOGICAL*1 TERM(6)
C
C     ASSIGN THE TERMINAL
C
      WRITE (5,554)
554   FORMAT (1H ,'OUTPUT TO ?')
      READ (5,510) (TERM(I),I=1,6)
510   FORMAT(6A1)
      CALL ASSIGN(1,TERM)
C
C     ASSIGN THE OPERATOR M
C
100   WRITE (5,520)
520   FORMAT (1H ,'ENTER M0,M1,M2')
      READ (5,530) SIG0,SIG1,SIG2
530   FORMAT (F,F,F)
C
C     ASSIGN THE SERVO TYPE
C
      WRITE (5,521)
521   FORMAT (1H ,'ENTER DEADBAND, POSN LIMIT,RATE LIMIT,POLE,DIST
      READ (5,522) DEAD,POSLIM,RATLIM,POLE,DIST
522   FORMAT (F,F,F,F,F)
      RATLIM=0.003125*RATLIM
      C=-1.0*EXP(-0.003125*POLE)
      GNS=1.0+C
C
C     ASSIGN THE RATE TYPE
C
      WRITE (5,550)
550   FORMAT (1H ,'1=ESTIMATED RATE, 0=TRUE')
      READ (5,551) ISWICH
551   FORMAT (I1)
C
C     ASSIGN THE GAINS
C
```

ZROLL.FORTRAN Listing cont.

```
104       WRITE (5,532)
532       FORMAT (1H ,'ENTER K, DELTA K,FIXED PART')
          READ (5,531) FIXK1,FIXK2,FIXK3,DELTA1,DELTA2,DELTA3,FIXK
531       FORMAT (F,F,F,F,F,F,F)
C
C         SET UP STORED VALUES, AND INSTALL 40M/S PARAMETERS
C
          GNE=0.000731193
          A=0.979513
          B=-0.940559
          GNR=0.4631608
          GNA=152.8
          XE11=0.030028
          XE1=0.030028
          XE21=0.5051747
          XE2=0.5051747
105       XS1=0.0
          XS=0.0
          XS0=0.0
          XR1=0.0
          XR=0.0
          XA1=0.0
          XA=0.0
          SVC=0.0
          ER1=1.0
          ER2=1.0
          ERROR=1.0
          RATE=0.0
          ACCEL=0.0
          I=8
          J=0
C
C         START OF SIMULATION
C
          DO 110 K=1,1280
          IF(I.NE.8) GO TO 126
          J=J+1
C
C         EVALUATE SIGMA
C
          IERROR=2048*ERROR
          QERROR=IERROR/2048.0
          ERATE=40.0*(1.5*QERROR-2.0*ER1+0.5*ER2)
          IF (ISWICH.EQ.0) ERATE=RATE
          SIGMA(J)=SIG2*ACCEL+SIG1*ERATE+SIG0*ERROR
          ER2=ER1
          ER1=QERROR
C
C         THE SWITCHING LOGIC
C
          IF((SIGMA(J)*ERROR).GE.0.0) GO TO 111
          GAIN1=FIXK1-DELTA1
          GO TO 112
111       GAIN1=FIXK1+DELTA1
112       IF((SIGMA(J)*RATE).GE.0.0) GO TO 113
          GAIN2=FIXK2-DELTA2
          GO TO 114
```

ZROLL.FORTRAN Listing cont.

```
113       GAIN2=FIXK2+DELTA2
114       IF((SIGMA(J)*ACCEL).GE.0.0) GO TO 115
          GAIN3=FIXK3-DELTA3
          GO TO 116
115       GAIN3=FIXK3+DELTA3
116       IF((SIGMA(J)*ERROR).GE.0.0) GO TO 130
          GAINS(J)=FIXK
          GO TO 131
130       GAINS(J)=-1.0*FIXK
131       SVDEM=GAIN1*ERROR+GAIN2*RATE+GAIN3*ACCEL+GAINS(J)
          IF(ABS(SVDEM-SVO).LT.DEAD) SVDEM=SVO
          SVO=SVDEM
C
C         THE NON LINEAR SERVO MODEL
C
126       IF (ABS(XS-XSO).LT.RATLIM) GO TO 127
          SIGN=1.0
          IF ((XS-XSO).LT.0.0) SIGN = - 1.0
          XS=XSO+RATLIM*SIGN
127       IF (ABS(XS).GT.POSLIM) XS=POSLIM*XS/(ABS(XS))
          XSO=XS
C
C         THE STATE EQUATIONS
C
          XS1=GNS*SVDEM-C*XS
          XE11=GNE*(XS+DIST)+XE1
          XE21=XE1-B*XE2
          XR1=GNR*(XS+DIST)-B*XR
          XA1=GNA*(XS+DIST)-B*XA
C
C         THE OUTPUT EQUATIONS
C
          ERROR=XE1+(A-B)*XE2
          RATE=XR
          ACCEL=GNA*XS-(1.0+B)*XA
          IF (I.NE.8) GO TO 128
          SERVO(J)=XS
          ROLL(J)=ERROR
          ROLDOT(J)=RATE
          TIME(J)=(K/320.0)-0.003125
128       XS=XS1
          XE1=XE11
          XE2=XE21
          XR=XR1
          XA=XA1
          I=I+1
          IF(I.EQ.9) I=1
110       CONTINUE
          ROLL(160)=-0.2
          ROLL(159)=1.2
```

```
C
C       OPTION MENU
C
        CALL SETBAS
117     WRITE (5,570)
570     FORMAT (1H ,'0:M0,M1 1:22M/S 2:50M/S 3:ROLL 4:SERVO')
        WRITE (5,580)
580     FORMAT (1H ,'5:K,DELTA 6:SIGMA 7:STOP 8:SIGPLANE 9:SAVE')
        READ (5,540)' IANS
540     FORMAT (I1)
        IF(IANS.EQ.0)GO TO 100
        IF(IANS.EQ.1)GO TO 101
        IF(IANS.EQ.2)GO TO 102
        IF(IANS.EQ.3)GO TO 118
        IF(IANS.EQ.4)GO TO 124
        IF(IANS.EQ.5)GO TO 104
        IF(IANS.EQ.6)GO TO 119
        IF(IANS.EQ.7)GO TO 121
        IF(IANS.EQ.8)GO TO 120
        IF(IANS.EQ.9)GO TO 125
        GO TO 121
118     CALL PLOT(TIME,ROLL,160,1,,,.TRUE.,,.TRUE.,,.FALSE.,,
     1  'ROLL STEP RESPONSE $','TIME, SEC.$',
     1  'OUTPUT, RAD.$')
        GO TO 117,
119     CALL PLOT(TIME,SIGMA,160,1,,,.TRUE.,,.TRUE.,,.FALSE.,,
     1  'SIGMA $','TIME SEC $','SIGMA $')
        GO TO 117
120     CALL PLOT(TIME,GAINS,160,1,,,.TRUE.,,.TRUE.,,.FALSE.,,
     1  '(RATE $','TIME SEC $','RATE $')
        GO TO 117
124     CALL PLOT(TIME,SERVO,160,1,,,.TRUE.,,.TRUE.,,,FALSE.,,
     1  'SERVO DISPLACEMENT $','TIME SEC $','RAD $')
        GO TO 117
125     CALL SAVE
        GO TO 117
101     GNE=0.000220375
        A=0.9879341
        B=-0.967297
        GNR=0.1401564
        GNA=45.61
        XE11=0.0164507
        XE1=XE11
        XE21=0.5030347
        XE2=XE21
        GO TO 105
102     GNE=0.00113141
        A=0.9745825
        B=-0.926614
        GNR=0.7149042
        GNA=237.6
        XE11=0.0371693
        XE1=XE11
        XE21=0.5064361
        XE2=XE21
        GO TO 105
121     STOP
        END
```

APPENDIX G

ANALOGUE COMPUTER MODELLING OF THE AIRCRAFT TRANSFER FUNCTIONS.

The hybrid simulation results presented in Chapter 7 are for Stabileye Mk 1. This is now obsolete so the material given here will be for the later Mk 3 airframe. The calculations follow the method suggested in reference (G1). The A.T.F's for 50m/s will be used as an example, with the potentiometer settings for 37.5m/s and 25m/s provided without their derivation.

G.1 PITCH.

The aircraft transfer function in s, is:

$$\frac{\theta}{\delta e} = \frac{-109.56(s + 5.173)}{s(s^2 + 10.827s + 162.83)}$$

Introducing the, dummy variable, z gives:

$$\frac{-\theta}{z} = 10.956s + 56.573 \quad \text{and} \quad \frac{z}{\delta e} = \frac{10.0}{s^3 + 10.827s^2 + 162.83s}$$

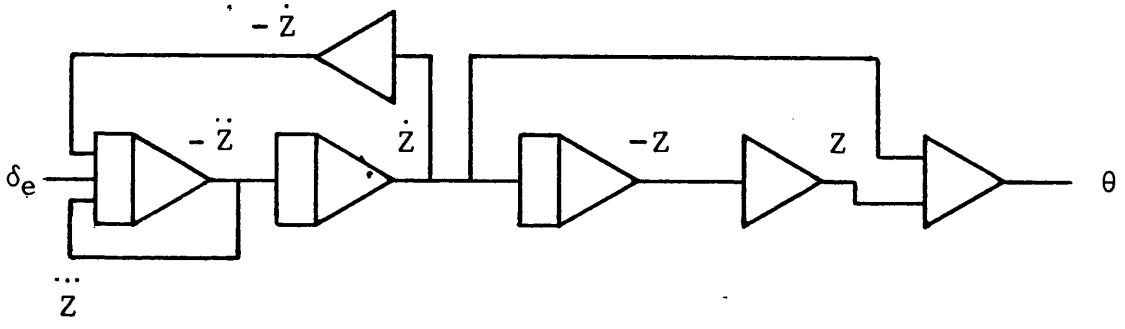
(Note, this is not the complex variable z, used in the sampled data analysis.)

Taking inverse transforms gives:

$$\dddot{z} + 10.827\,\ddot{z} + 162.83\,\dot{z} = 10.0\,\delta e$$

and $$-\theta = 10.956\,\dot{z} + 56.673\,z$$

So the un-scaled flow diagram, drawn by composing the highest derivative is:



The decomposed equations are:

1) $\frac{-d}{dt}(-\dddot{z}) = 10.0\,\delta e - 10.827\,\ddot{z} - 162.83\,\dot{z}$

2) $\frac{-d}{dt}(\dot{z}) = -\ddot{z}$

3) $\frac{-d}{dt}(-z) = \dot{z}$

4) $-(-\theta) = 10.956\,\dot{z} + 56.673\,z$

Physical scaling: The elevator is limited to +/- 15°, or +/- 0.26 rad. The pitch attitude is limited to +/-8° , or +/- 0.1396 rad. The scaling is arranged so that the output from the model mimics that of the gyroscope, i.e. 16.67°= 5.0 V, so 33.3° = 10.0 V (0.582 rad). Therefore the physical scaling is chosen as:

$\frac{\delta e}{0.3}$	$\frac{\theta}{0.582}$	$\frac{z}{0.01}$	$\frac{\dot{z}}{0.02}$	$\frac{\ddot{z}}{0.1}$

Hence the scaled equations are:

1) $\frac{-d}{dt}\left(\frac{-\ddot{z}}{0.1}\right) = \frac{10.0 \times 0.3}{0.1}\left(\frac{\delta e}{0.3}\right) - \frac{10.827 \times 0.1}{0.1}\left(\frac{\ddot{z}}{0.1}\right) - \frac{-162.83 \times 0.02}{0.1}\left(\frac{\dot{z}}{0.02}\right)$

2) $\frac{-d}{dt}\left(\frac{\dot{z}}{0.02}\right) = \frac{-0.1}{0.02}\left(\frac{\ddot{z}}{0.1}\right)$

3) $\frac{-d}{dt}\left(\frac{-z}{0.01}\right) = \frac{0.02}{0.01}\left(\frac{\dot{z}}{0.02}\right)$

4) $-\left(\frac{-\theta}{0.582}\right) = \frac{10.956 \times 0.02}{0.582}\left(\frac{\dot{z}}{0.02}\right) + \frac{56.673 \times 0.01}{0.582}\left(\frac{z}{0.01}\right)$

The machine equations are: (Note that the no. 10 outside the square parentheses inplies a "nose" gain of 10 on the integrator block.)

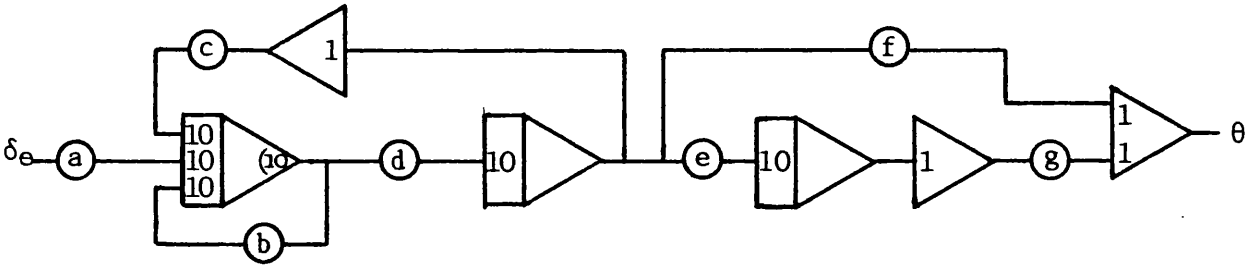
$$1)\ \frac{-d}{dt}\left(\frac{-\dot{z}}{0.1}\right) = 10\,(0.30)\,10\left(\frac{\delta e}{0.3}\right) - 10\,(0.108)\,10\left(\frac{\dot{z}}{0.1}\right) - 10\,(0.326)\,10\left(\frac{\dot{z}}{0.02}\right)$$

$$2)\ \frac{-d}{dt}\left(\frac{\dot{z}}{0.02}\right) = -(0.50)\,10\left(\frac{\dot{z}}{0.1}\right)$$

$$3)\ \frac{-d}{dt}\left(\frac{-z}{0.01}\right) = (0.20)\,10\left(\frac{\dot{z}}{0.02}\right)$$

$$4)\ -\left(\frac{-\theta}{0.582}\right) = (0.376)\left(\frac{\dot{z}}{0.02}\right) + (0.974)\left(\frac{z}{0.01}\right)$$

The machine diagram is:



The coefficients are:	50m/s	37.5m/s	25m/s
a	0.3	0.3	0.3
b	0.108	0.812(1)	0.541(1)
c	0.326	0.183	0.145
d	0.5	0.5	0.5
e	0.2	0.2	0.2
f	0.376	0.212	0.094
g	0.974	0.406	0.122

G.2 ROLL.

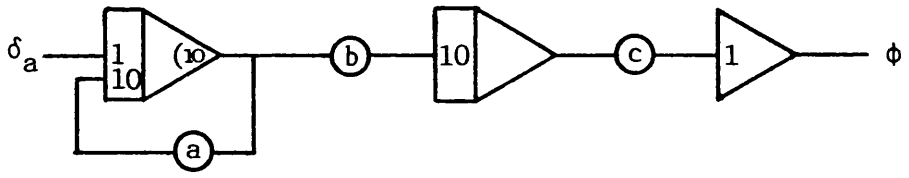
The aircraft transfer function in s, is:

$$\frac{\phi}{\delta a} = \frac{-237.6}{s(s+24.39)}$$

Physical scaling: The aileron is limited to +/- 10°, or +/- 0.1745 rad. The gyroscope produces +/- 5.0 V = +/- 80°, so +/- 10.0 V = +/- 160°, or +/- 2.79 rad. Therefore the physical scaling is chosen as:

$\frac{\phi}{}$	$\frac{\delta a}{}$	$\frac{z}{}$	$\frac{\dot{z}}{}$
2.79	0.2	0.1	0.2

By the same process as for the pitch transfer functions, the machine diagram can be derived as:



The coefficients are:	50m/s	37.5m/s	25m/s
a	0.244	0.182	0.122
b	0.2	0.2	0.2
c	0.852	0.476	0.209